AD660043

SOME RESULTS IN A THEORY OF PROBLEM SOLVING

R. B. BANERJI

SRC 126-A-67-59

BEST

AVAILABLE

COPY

SYSTEMS RESEARCH CENTER

CASE WESTERN RESERVE UNIVERSITY

UNIVERSITY CIRCLE

CLEVELAND, OHIO 44106

NOVEMBER, 1967

This work was sponsored in part by the U S A Air Force Office of Scientific Research under grants AF-OSR-125-67, AF-OSR-125-65, and by the National Science Foundation under grants GK 1386, GK185 and GP 658.

TABLE OF CONTENTS

FORWARD

This report constitutes the first 3 chapters of a book on Artificial Intelligence which will hopefully be published next year. Most of the next 3 chapters of the book are presently available in the form of previous reports and publications and hence will not be published as a report. This report is being published jointly with the RCA Laboratories.

The work reported on here was carried out by myself and graduate students supported by the U S Air Force Office of Scientific Research under grants AF-OSR-125-67, AF-OSR-125-65, and by the National Science Foundation under grants GK 1386, GK 185 and GP 658. Some of the graduate students whose work is reported here were supported by NDEA, NSF and Case Fellowships.

Most of the present report was written while the author was in the RCA Laboratories in Princeton where his time was partially supported by the U S Office of Scientific Research under contract no. AF49(638)-1184. Dr. Saul Amarel of the RCA Laboratories is to be thanked for his kind cooperation and many comments.

CHAPTER I - INTRODUCTION

1. The Field of Endeavor

The field commonly called Artificial Intelligence may, perhaps, be described as "The totality of attempts to make and understand machines that perform tasks which till recently only human beings could perform and perform them with efficiency and effectiveness comparable to a human." There has been a large amount of controversy on the aptness of the name "Artificial Intelligence" for the field. There is probably some utility to this kind of discussion in view of the general impression the name makes on the mind of the lay public. However, for the purposes of technical discussion one may decide not to attach any significance to the name apart from what is implied by the definition. This kind of special technical use of well-known words is not without precedence. One may recall the word "Energy" as used in Physics or "Group" as used in Modern Algebra.

The definition of the field given above is certainly not very precise. It has always been extremely difficult to define areas of technical endeavors with precision. However, it may be worthwhile to try and make some clarifying remarks which seem called for.

In the definition, when one refers to an activity performed by a human being, it is not clear as to what aspects of the activity are considered important. If a machine is designed to play checkers, for instance, one can demand that:

i) it wins often against human players,

ii) it produces electroencephalograms similar to humans engaged in playing checkers,

iii) it holds and moves pieces on a cherkerboard with the same grace as some human beings do,

iv) it makes a move is less than ten minutes.

It seems to have been the consensus of the practitioners in the field that the first and fourth demand above should be made while the second and third demand should not be made. There is indeed some rationale for such a consensus. However, the discussion of the rationale will carry this book far from its purpose as stated in the preface. For the purposes of this book any task under discussion will be considered to be as described by the consensus. No effort will be made to justify the consensus.

To avoid a certain unfortunate implication of the definition another aspect of the definition will also bear clarification: the interpretation of the phrase "till recently". If the phrase is interpreted to refer to the recent past at any future time of discussion(within the period of relevance of the book), then the field of Artificial Intelligence takes on an aspect of ephemerality and becomes a clearing house for ill-understood techniques. This is not intended and the phrase "till recently" should be interpreted to refer, within the period of relevance of this book, to the time of writing of the book. Hence when a machine is made to perform a human activity for the first time, the definition should not be taken to assert that later attempts to make better versions of the machine should be considered as outside the field of Artificial Intelligence. Even after a machine is constructed which meets the specification of the definition, any attempts to make machines which perform the same task by a different method would still be in the field. So would be theoretical attempts to shed light on the performance of such machines, since this might facilitate the construction of machines which perform other similar tasks.

However once the method of construction of machines for the performance of a certain class of human tasks is well understood, the

construction of such machines would also be considered as an activity in some other field depending on the nature of the machine as well as the purpose of building it.

The last point needs discussion. A rather cursory glance through the activities in the field of artificial intelligence will reveal a number of different techniques and purposes.

There have been some attempts made at using digital computer programs for finding satisfactory solutions of industrial or engineering design problems where optimal solution was either hard to define or too time consuming to obtain. If techniques for writing such programs for specific purposes become well-understood and perfected, these activities would probably be considered as parts of the appropriate branches of management or engineering - or disappear altogether, being replaced by sub-routine libraries.

There have been many digital computer programs designed [1] to simulate certain activities of the human mind. There have been simu-lations of groups of humans, as in Sociological phenomena (including economic phenomena). There have also been simulations of individual humans finding solutions of complex combinational problems, making deductions from a corpus of given facts or recalling facts by association. Once the method of design of such programs and their use for Psychological or Sociological investigation is well understood, it may again be reasonable to classify such activities in Psychology or Sociology. As before, Economics has been included in Sociology for the purposes of this discussion.

The state of formalization of the field, however, is such that it is difficult to say with respect to a specific attempt, as to whether it is an effort to simulate human processes or an effort to solve a certain

problem. If the problem is one whose solution is of immediate applicability in technology management or, one can even ignore any purposes the attempt serves for the Psychologist or Sociologist and classify the attempt as potentially belonging to technology or management. If, however, the problem is of no immediate applicability, it would be unfair to classify the attempt as belonging to Psychology or Sociology unless the motivation for writing the program arose from a psychological or Sociological interest. Often such programs are claimed to have been designed to simulate the way the programmer believes he would attempt to solve the problem himself; however, the motivation in these cases may come from a desire to solve the problem rather than to understand human phenomena.

There has been considerable discussion as to whether such attempts need be classified as activities in a recognized field of Science. Many feel that it may be easier to classify them as recreational activities of some clever computer programmers. However, there have been many occasions in the history of Science where the recreational activities of some people have led to insights that have enriched Science or even technology or business; these computer activities may well lead to such an enrichment. Study of the literature in the field indicates that the various attempts made at writing efficient programs for finding solutions of large combinational problems show some basic communality of approach and technique. Semi-formal attempts have also been made to codify these similarities into a theory. Such study of empirical attempts and results, together with attempts to unify them is becoming an important branch of Artificial Intelligence. One may call this branch, "Theory of Problem Solving". It appears that the time is not too far distant when this kind of activity - a study of problems and their solutions independent of any psychological

connotations - will denote a well defined area of endeavor. Apart from the appropriateness of this field as an important subfield of artificial intelligence, it may also be considered as a branch of Computer Science or perhaps mathematics. In a less formal way, this activity has followed by students of Methodology in Philosophy.

By its very nature, theory of Problem Solving is an applications-oriented discipline. Even at this early stage of its development, technique and ideas originated in the field of Artificial Intelligence has found and promises to find fruitful applications in Science and Technology [2].

"Pattern Recognition" is often considered as a separate branch of Artificial Intelligence, although there has been a growing recognition over the years of the close relationship between this field and what has been delineated above as "Theory of Problem Solving". However, this relationship is very ill-understood. One of the reasons for this is the lack of a clear set of definitions of terms used in the field of "Pattern Recognition". There has been enough activity in the field to indicate that the basic idea deals with the recognition of a given object as belonging to a given set of objects. This recognition is only possible when there is a statement (in some language) which is true for all objects in the given set and is false for all objects not in the given set . For reasons associated with the history of the work at Case, the set of objects will be called the "pattern" or "concept". This is at slight variance with the intuitive use of the term "pattern." One often uses the term to denote the description of the set or those statements about the recognizable object which implies the description. For the purposes of the book, the words "pattern" and "concept" will stand for the set of objects.

A perusal of the usages in the field indicates that a "pattern recognizer" is a machine which can form the description of a pattern when presented with a small number of objects in the pattern. The term "pattern learning" will be used in this book for the activity, reserving "recognition" for the much simpler activity of recognizing whether an object belongs to a pattern with a given description.

While most activities in the other fields of Artificial Intelligence have been carried out with the aid of digital computers, a considerable amount of the work in Pattern Recognition has used the aid of other devices. The use of adaptive threshold logic elements was one of the first steps taken in this field of endeavor. By now the original uproar regarding the neuro-physiological significance of such devices has subsided. However, threshold logic (adaptive or otherwise) remains an interesting area of study in the field of switching theory. It is possible that a theory of "neural networks" based on such devices will have a strong influence on the theory of pattern recognition; however, such a possibility seems remote at present.

In what has gone above we have made an attempt to subdivide the field of Artificial Intelligence. Almost the entire content of this book deals with the area designated as "Theory of Problem Solving". Since Pattern Recognition (studied as a computer algorithm) is very closely related to this area, pattern recognition will also be discussed at length.

The approach that will be used may be described as Systems Theoretic. A model for problem situations will be set up using certain abstract and quite elementary set theoretic concepts. In its abstract form, such a model can be looked upon as a generalized definition only; the model does not appear to contain indications of what might be considered

to be methods of solving the problem. To obtain such indications, certain further structures would have to be assumed. Stating the matter another way, one may say that the minimal structure needed for defining a problem is not sufficient to define methods of solution. Various forms of extra structures can be introduced as tools for the discussion of methods of solution. In this book only one such structure has been chosen. The reason for this choice is historical - in that this was the first structure that occured to the school of investigators whose work is presented here.

The resulting model, embodying the model of problems with certain extra structures, is almost identical with the model of problems envisaged in the General Problem Solver developed by Newell, Simon and Shaw. However although the model as it stands is sufficient for the beginnings of a discussion of solution methods, this advantage was not used by the originators of the General Problem Solver. Instead, a specific method of solution was developed and studied, but never described with adequate precision.

It has been argued that the mere existence of an abstract model for solution methods is of no value. What is crucial is an adequate description of the problem which makes it amenable to the solution method. The argument is perfectly valid in so far as it says that abstract sets do not have sufficient structures for the study of any specific solution method. However, the argument does not imply (as it is often made to imply) that one therefore should not use a precise theoretical approach to describe solution methods. This false implication has led to the use of intuitive and imprecise descriptions of solution methods. It is hard to make any judgement as to whether this has been of advantage or of disadvantage to the field. However at the present state of the art an effort at

making one's discussions precise and mathematically correct promises immediate returns; in case of communication and documentation of ideas and results, in quick evaluation of basically erroneous ideas - and perhaps also in aiding innovations by interaction with related fields.

If one considers the major part of the argument against precise models of solution methods, one is forced to agree that a problem formulation to be meaningful must have with it an adequate representation of the problem in some language. This is in no way at variance with the basic tenets of systems theory. It is clear that no specific problem can be formulated unless the sets associated with the problem are adequately described in some language. Indeed, the effectiveness of this language of description turns out to be easy to discuss in terms of its efficiency in describing the sets associated with the solution methods. But this needs precise definitions of the associated sets.

The above discussion indicates another important belief on which this book is based. A meaningful theory of problems and their solution should include or have close relationships with a theory of descriptions and description languages. Such a theory will be discussed in this book, together with a model for problems and some models for problem solution.

It is not claimed here that either the models of problems and solutions or the theory of descriptions as they stand at present are adequate for the purposes of Artificial Intelligence. However, a belief is inherent that any meaningful theory of problem solving must include such precise models and theories.

2. Outline of the Basic Models

The present book will deal with models of problems and two-person games. Both of these will be specialization of a general model discussed by Marino [3]. This model can be looked upon as a general model of control systems in addition to that of problems and games.

Basically, one is given a set of objects which are called "states" in control theory and may be called "situations" in the theory of problems and games. In addition, there are two other sets, whose elements will be called "controls" and "disturbances". Given a control paired with a disturbance, certain situations are changed to other situations. The modes of such changes are pre-specified. A certain set of situations have their elements labelled as desirable or "winning situations". Given a situation, the control problem is generally stated as the problem of finding a control such that no matter what disturbance it is paired with, the resulting situation is a desirable one.

When a real control problem is posed in such abstract terms one often finds that the set of controls and the set of disturbances are so intractable that an appropriate control is practically impossible to choose among the host of possibilities. Fortunately, most real problems impose certain extra properties on the situations, controls and disturbances. Many have the property that the control and the disturbance sets are "generated" by a more tractable set of elements. This will be made precise later. For the present purpose one may say roughly that each control is a sequence of "elementary" controls and each disturbance is a sequence of "elementary" disturbances. The problem then reduces to that of finding a sequence of elementary controls such that no matter what elementary disturbance is paired with each elementary control, the final result of the

sequence of pairs is a winning situation. This problem may be called the problem of finding an "open loop controller".

One difficulty often arises with such a specification of the problem - a difficulty which is often ignored in control problems but is of supreme importance in games. The difficulty arises because all elementary controls may not be applicable to all situations. As a result, a sequence of elementary controls cannot be chosen which will be applicable irrespective of what elementary disturbance they got paired with.

One can get around this difficulty by asking, not for a control sequence, but for a "control strategy". A strategy is an initial decision on the control to be used at each situation, any time the situation arises. Given a certain situation, one decides on the control dictated for the situation by the strategy. Depending on the disturbance that is paired with this control, a new situation arises. A new control is then dictated for the new situation by the strategy and the process is repeated. If such a sequence ultimately results in a winning situation irrespective of the disturbance, the strategy is called a "winning strategy". The finding of a "winning strategy" is analogous to finding a "closed loop controller".

The idea of a strategy essentially envisages a Bellman-type embedding of a problem in a larger problem [4]. It is of advantage even in cases where it is inessential - that is, when the applicability of controls are independent of the situation. Moreover, it is well-known that in some control problems one cannot build an open loop controller while a closed loop controller can be built.

The general model of control situations can be specialized to yield some special classes of the so-called problem-situations. Mesarovic [5]

has classified problems into various types one of which is as follows: "Given a set S, subsets T and H of S and a set of functions F such that each element of F maps S into S, to find a member of F which maps each element of H into some element of T".

When one defines a real problem in this framework, one faces the same kind of difficulty as in the case of the control problems. Mesarovic had pointed out in his paper that the set F, to be tractable, should be "constructively defined". Windeknecht [6] assumed a specific constructive structure of F by assuming that elements are obtained by composing functions from a finite set F_o of functions. He also stipulated that the elements of F_o were partially defined over S, so that the composition operator defined a partial semigroup rather than a semigroup. H was considered as a unit set. This model will be followed in this book except that for the purposes of this book it will not be assumed that F_o is a finite set.

If a problem is defined in the manner given in the above paragraph, one sees a clear relationship between this model of a problem with the model formulated by Marino. If in the model of Marino the set of elementary disturbances be a unit set (whose element may be called "inaction") then each elementary control defines a map from situations to situations and can be taken to be members of F_o. This analogy will be pursued rigorously in the next chapter.

The model of problems discussed here is also very closely related to the model used in the General Problem Solver. In this latter model one is given a set of situations and a set of transformations each of which changes some situations to some other situations. One is supposed to change a given situation into another given situation (the "goal") by

applying a sequence of transformations. The point at which the model of the general problem solver differs from the model envisaged in this book lies in the introduction of a set of goals (winning situations) instead of a single one. This difference is not merely a matter of generalization. It will be shown in the next chapter that in some of the specific cases handled by the general problem solver, one is actually interested in a set of goals rather than a single goal. This has become evident especially in view of some recent extensions [7] of the original GPS.

At this point no attempt will be made to discuss the major part of the General Iroblem Solver which deals with methods for finding solutions to problems. This will be done later. For the purpose of the present section, it would be more important to point out how the model proposed by Marino can be reduced to the model of a two-person game.

If the extended form of a zero-sum two person game of the von Neuman Morgenstern type [8] be restricted to have pay-off functions whose values are only 1, -1, then such games can readily be shown to be representable by a special class of Marino-type models. In these, one fixes a specific elementary control and a specific disturbance, each called "inaction". It may then be specified that in each situation either the control inaction or the disturbance inaction is applicable, but not both. This introduces the concept of the player's move and the opponent's move. Also a further axiom can be introduced if necessary forcing the player and the opponent to move alternately.

The above model can be made to represent an N-person game in that the disturbances may be considered to be the result of the joint action of n-1 players. However, since such an assertion sheds no light on the behaviors of the separate n-1 players (with respect to coalitions and related phenomena), this assertion will not be made seriously here.

However, the model here is not so specific as not to include games with incomplete information. One need not construe the elements of S as embodying the entire information regarding the past of the game. As a matter of fact it will be noticed that in the present model the entire past is not embodied in a situation. Unlike in von-Neuman's model of extended games, the present model is not a tree, but an automaton (or a labelled directed graph). One can carry this process a step further and consider a situation as the "state of information" of a player, i.e. a subset of the set of "actual situations". It would not be too difficult to show how a game with incomplete information can be converted into a "larger" game (with a larger number of situations) with complete information - in a way analogous to converting a non-deterministic automaton to a deterministic automaton. However, this generalization does not shed any further light on the methods of solution in the abstract, and will not be pursued further in this book.

3. A Set-Theoretic View of Pattern Recognition

The main purpose of this book will be to consider certain methods for finding solutions to problems and games and their relationship to pattern recognition. This relationship can be discussed in a clear manner if both the activities (problem-solving and pattern recognition) can be discussed within the same mathematical framework. As has been said before, the framework of elementary set theory will be used in this book.

In their essence, the methods of problem solving will be taken to stem from the existence of certain basic subsets of S associated with a problem or game. Some such sets (like the winning situations, or the domain of applicability of different controls, etc.) are provided by the rules of the game itself. Certain others are suggested by the idea of a solution. To see this roughly (detailed discussions will appear later) one can imagine that one person who knows the solution of the problem for every initial situation intends to transfer to someone else his knowledge. For every specific control he will have to define the set of situations in which that control is to be used. These are one class of sets associated with the idea of solution. Other sets associated with the idea of solutions will be considered in the next chapter. Meanwhile, it is crucial to make the point that in all cases of interest the set S of situations is extremely large. Hence those sets cannot be exhibited by any practicable enumeration technique. It is this difficulty which holds up efforts at problem solving.

However, the difficulty may not be unsurmountable. It may be recalled that although the set of situations is large, no difficulty arises about its enumeration. Any chess player can recognize a chess

position as a chess position. Similarly, the set of all mates never have to be enumerated either - a mate is easily recognizable when it occurs. The rules of the game give us the controls, disturbances and winning situation, not as enumerations but as "descriptions" - methods by which members of these sets can be recognized when they arise. Similarly any solution method, to be practicable, must be expressed in terms of the descriptions of the sets associated with the solution methods.

The difficulty lies with the word "practicable". The practicability of a strategy is strongly dependent on the language one uses for the description of the sets associated with the solution methods. One can change the language of description to change the practicability of various solution methods.

The difficulty of finding a solution of a game/or problem lies in the fact that the language which is needed for practicable descriptions of the sets associated with the solution method is seldom identical to the one used in describing its rules, i.e. the controls and the winning, losing and draw situations. Ideas regarding description languages is crucial here - as they are in any adequate theory of pattern recognition. In what follows, an approach to the formal definition of such terms as "description", "description language", "pattern", etc. will be given.

As stated previously in Section 1, a pattern may be defined as a set of objects. One can consider the pattern of all the letters "A" projected on an array of photo-cells, the pattern of all checker positions showing satisfactory center control [9], the pattern of all sets of theorems from which a desired theorem can be obtained by Modus Ponens [10].

It will be worthwhile to realize at the outset that when one tries to develop a language for describing a class of patterns, one cannot

seriously mean to be able to describe the class of all patterns (if one tries, one faces immediate dilemmas like, "Do the set of all paterns which are not elements of themselves form a pattern?"). The class of patterns has to be restricted. The initial restriction that will be made here will be to a class of subsets of a given set, which we shall call the Universe of discourse or simply the Universe. By definition, any object will be taken to belong to the Universe.

If the universe is finite, one can consider any subset of it to be described by a list of its elements; but if the subset (or pattern) is large, one cannot call such a description practicable. One has, at this point, to make some further restrictions - to assume some further structure for the universe.

Without too much loss of realism, one can make the assumption that there are certain general statements one can make about elements of the universe whose truth can be tested easily for any specific element of the universe. Such statements will be called "Predicates" in keeping with literature in Symbolic Logic [16]. The assumption will be that in addition to the universe, one is also given a set of predicates.

It has already been indicated that the description of a pattern yields a procedure which has the following property; given an element of the universe, the application of the procedure determines whether the element belongs to the pattern or not. Clearly any pattern, each of whose elements satisfy a given predicate (which, in turn, is satisfied only by the elements of the pattern) is describable by that predicate. One can say, therefore, that our assumption has led us to a class of patterns which are "easily describable" in the sense that their descriptions are embodied by single predicates whose truths are easily tested for.

We can take the easily describable class of patterns as forming the generators of a Boolean Algebra. The class of describable patterns may be restricted to the elements of this Boolean Algebra. If one does this, one may make the rather trivial statement that a description language which incorporates the initial predicates and uses the logical connectives of "or", "and", "not", "implies", etc. will be able to describe any element of the class of patterns under consideration.

The major problem, however, is not so much of the possibility of description, but of the efficiency of description. One needs descriptions where the elementary predicates are combined in such a way that the resulting statements are not inordinately long. Also, one needs the statements to be such that their truth and falsity can be tested for without an inordinate amount of processing. This once more restricts the class of easily describable patterns.

Logical connectives are not the only means by which the initial predicates can be combined. A large amount of work has gone into combining predicates by threshold gates [17], for instance. The patterns which yield short descriptions through single applications of threshold logic form a sub-class of the class of all describable patterns. All describable patterns can be described by more than one application of threshold logic.

One can in an informal way define a description language to consist of a set of initial predicates and a set of connectives or modes of combination which can be used to combine the initial predicates to yield descriptions of describable patterns. The class of patterns easily describable by a given description language depends on the description language.

The class of patterns whose elements are to be described is determined by the problem which necessitates the recognition of elements of the patterns in the class. The basic problem then reduces to the following, "Given a class of patterns, to develop a description language which yields short and easily processable descriptions for all patterns in the class."

At the present time no practicable method for the solution of such a problem has been developed (as a matter of fact solution methods for very few problems have been developed). However, a study of the problem in its formal aspect indicates the need for a uniform model of description languages in which different description languages can be embedded. This enables changing one language to another - a definite necessity for the specification of the basic problem itself. In what follows some of the basic building blocks for a generalized description language of this kind will be specified.

Initially, some structure will be assumed for the predicates of the language. It will be assumed that each specifies the result of a test performed on an element of the universe. In effect, the test is a mapping from the universe to the set of results. The kernel of this map (an equivalence relation) induces a partition on the universe. The elements of this partition are mapped one-to-one onto the set of results. The image of each element of the partition under this map will be called the "name" of the element.

One can thus make two equivalent statements about an element u of the universe: (1) "The result of performing test P on u is p_i" or (2) "u belongs to the element p_1 of the partition P". For historical reasons the second form of the statement will be adhered to.

With the basis made above one can define a pattern as either an element of a partition or obtained from other patterns by set theoretic operations. In what follows each partition will be called an input property and the elements of a property will be called its values.

Any set of pairwise disjoint patterns whose union covers the universe will also be called a property. By definition, any input property is a property, but not conversely.

An object is a pattern which is either contained in or disjoint from any value of any input property. It can be shown quite readily that an object is either contained in or disjoint from values of all properties. It can further be seen that an object is an intersection of a class of values of input properties. They can, therefore be represented by a list of pairs of names, each pair consisting of the name of a value and the name of the property to which the value belongs. For example, a typical object might be "(P_1, p_{13}; P_2, p_{21}; P_3, p_{33})".

Any pattern can be described by a Boolean expression involving values or a statement involving predicates of the form $P(u) = p$ where P is an input property. The problem of finding the simplest expression describing a pattern is a problem closely analogous to finding the simplest expression for a switching function (as can be seen, switching functions are special cases - each input property has two values). The solution depends basically on what one means by "simple" and, as in the case of switching function, the solution can be found only for some restricted definitions of simplicity.

Very often, after the simplest description is found for a pattern, this "simplest" description is still so complex as to be unusable. At this point one can hope to find simpler descriptions if ones uses properties other than input properties in the description. To do this, one needs to

allow predicates of the form u ϵ K where K is the value of some property other than an input property. Of course, to use such a description one needs to invoke the description of K as a pattern. One thus obtains the analog of the Ashenhurst decomposition [18] of a Boolean function. In a later chapter some of the techniques and terminologies associated with these problems will be discussed.

The literature in the field of pattern recognition indicates that the only kind of switching functions that have found use in the field are those expressible as conjunctions and as threshold expressions. The largest effort in the field is spent on finding the "useful" properties that render the pattern expressible in one of these simple forms. Unfortunately, there is no uniform method for expressing the processes that yields values of these useful properties. If one considers the problem with the set-theoretical bias inherent in this book, a rather interesting uniformity emerges. One often finds that these processes yielding the values of the useful properties really process the names rather than the denotations of the input properties and their values.

There will be later occasions to discuss this kind of processing for some seemingly realistic situations. For the present one can consider the following rather artificial example, which is based on a rather well-known example used by Bruner [19].

Let the universe consist of the 9 configurations shown in Figure 1.1. We shall call the atomic objects of this universe 1,2,...9 for convenience, as shown in the figure. Formally, the universe will have two input properties, whose names will be "crosses" and "borders". (One could denote these properties by P_1 and P_2 in keeping with the definitions. However, since the future discussions will be heavily dependent on the names,

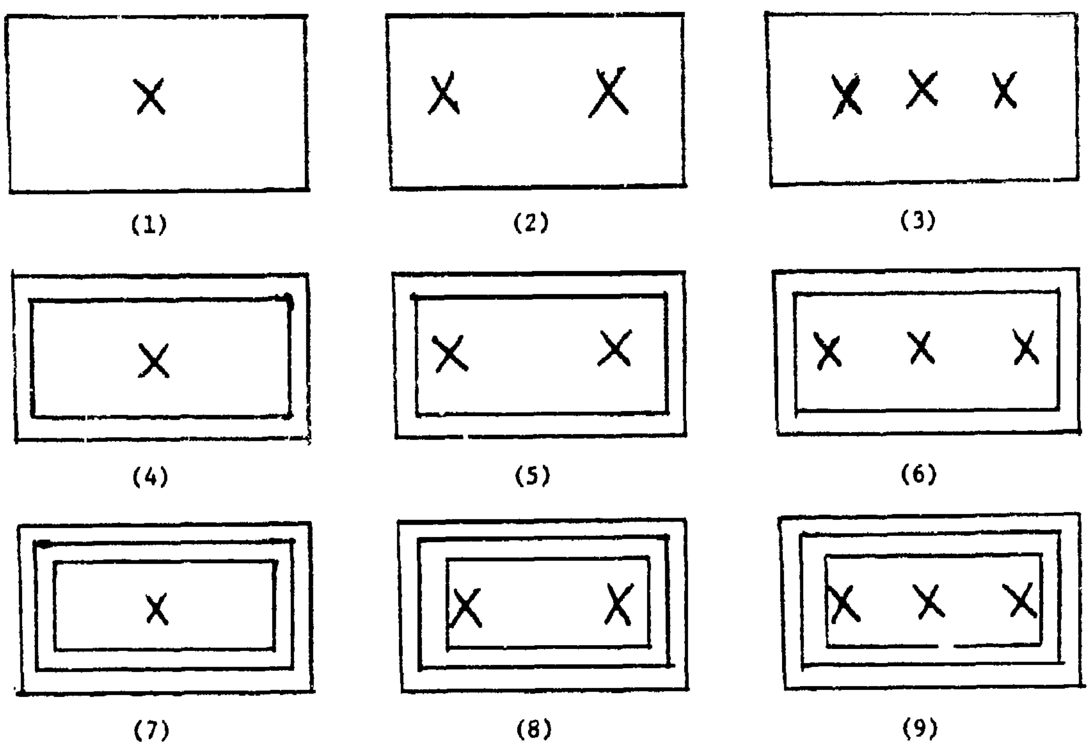


Fig. 1.1

A simple universe illustrating the use of non-input properties and the usefulness of processing names rather than denotations of values.

this may be as good a place as any to introduce names which have greater structure than abstract symbols. Each of the properties have three values which shall have "one", "two" and "three" as names.

To give intuitive meaning to these properties and values, let the set {1,4,7} be the value "one", of the property "crosses". Similarly let {4,4,6} be the value "two" of the property "borders".

This universe has the convenient property that every distinct atomic object is the member of a distinct and unique object. As a result, every subset of the universe is a pattern. For the purposes of the present discussion, the pattern {4,7,8} = A (say) may be considered. This set will be described in three different ways: one using only input properties, one by using properties other than input properties, one by using relationships between names of the values of properties. This will be done to illustrate the three methods, rather than to exhibit the difference in their efficiency as description methods. Such difference can be exhibited conclusively only for larger universes; examples will be given later when the description language is introduced formally.

One could describe the pattern A formally by the statement:

$$x \in A \equiv (\text{crosses}(x) = \text{one}) \wedge ((\text{borders}(x) = \text{two}) \vee (\text{borders}(x) = \text{three}))$$
$$\vee ((\text{crosses}(x) = \text{two}) \wedge (\text{borders}(x) = \text{three}))$$

involving only predicates of the type $P(u) = p$.

The pattern A could also be described with statements of the type $u \in K$ as follows

$$x \in A \equiv (x \in B) \wedge (x \in C)$$

$$x \in B \equiv (\text{crosses}(x) = \text{one}) \vee (\text{crosses}(x) = \text{two})$$

$$x \in C \equiv (\text{figures}(x) = \text{one}) \vee (\text{borders}(x) = \text{three})$$

This form of description is of advantage only if the patterns B and C can be involved in the description of many patterns other than A.

In many existing pattern recognition schemes, the switching function for a pattern is restricted to have some specific form. For instance it may be restricted to have a form realizable by a single threshold gate. If, for example, one makes the restriction that A be described by a minterm expression, the second description above would be according to such a restriction. The patterns B and C would be "useful" features for describing A. In most cases such features are obtained by processing the names of the values of the input properties and not in the form of a Boolean Expression as done here. It is desirable for the sake of uniformity and flexibility of the description language, however, to express such preprocessing of names in the same format as other descriptions.

To illustrate such a format a third alternative description for the pattern A above will be exemplified. This example will take advantage of the fact that the names of the values of the two properties, "crosses" and "borders", come from the set of numerals and the concept A can be described in English by saying "In any element of A, the number of crosses is less than the number of borders". The description language will need some method for expressing the relation, "less than". It will be shown later how a relation can be expressed as a pattern in the universe of ordered n-tuples. This involves the introduction of several new universes. For the present purpose, only one new universe needs be evoked. This will be evoked to enable the expression of the symbols "one", "two" and "three" (the names of the values of the properties crosses and borders) as their binary counterparts. Each numeral will have two properties, "head" and "tail", standing for the twos place and the ones place of their binary

expansion. The values of both these properties will be called F and T. The numeral "two" for example will take the form (head, T; tail, F).

With this new universe in mind one can describe the pattern A as follows

$$x \in A \equiv (\text{head}(\text{border}(x)) = 1) \wedge ((\text{head}(\text{crosses}(x)) = 0$$
$$\vee ((\text{tail}(\text{borders}(x)) = 1) \wedge (\text{tail}(\text{crosses}(x)) = 0)))$$

As before, the advantage of such a description becomes clear only in those cases where the universes involved are much larger. There will be later occasion to discuss this. It may, however, be pointed out at this point that even where such an advantage is obtainable, it is obtained at the expense of making the objects of the universe more complicated (imbuing the universe with greater structure). For example, as long as we used the symbols "one", "two" and "three" as values of crosses and borders, a typical object (6,say) of the universe would be (crosses, two; borders, three). When the values themselves are construed to come from the structured universe of binary numerals, the same object becomes (figures, (head,T;tail,F); borders,(head,T;tail,T)).

The relationship between the richness of the description language and the facility of problem solving will be discussed through some examples in later chapters. The present section can be brought to a close with the following remarks.

So far discussions have been limited to descriptions of patterns. Given a description language one wants to construct a processor which can operate on objects to determine whether they are contained in any given pattern. This presupposes certain restrictions on the language. For instance if the language is strong enough to describe recursively enumerable sets of objects, no processor of assured ability can be built.

Even assuming that one has a language which enables the construction of the corresponding processor, this would not solve the problem of "pattern recognition".

It appears from the literature that pattern recognition consists of forming rather than processing a description. What is generally envisaged as pattern recognition is the following phenomenon. A processor is presented with a set of objects, each of which is tagged to indicate whether or not is is contained in a given pattern. From this data, the processor is supposed to form a description (in some language) of a pattern which contains all the objects tagged as being contained in the given pattern and does not contain any object which is tagged as not being contained in the given pattern.

It is not overly difficult to build such a processor. The difficulty so far lies only the simplicity of the generated description. This has been discussed before. However, another demand is often made on the description generating processor. It is expected that the description constructed by it will be such that when an untagged new object is presented to the processor, it will fit the description if and only if it belongs to the given pattern. This is clearly an impossible task in general - the only evidence presented to the processor about the pattern having been the tagged objects. As long as the tagged objects do not exhaust all objects in the universe of discourse, one can always have a number of distinct patterns satisfying the tagging of the elements. The processor builds the description of only one of these patterns - it would be self-defeating to form all the descriptions, and even that would not help in the recognition of later untagged elements.

The phenomenon of "generalization" has received some attention from statisticians [20]. Their studies seem to indicate that the number of tagged objects needed for establishing a degree of confidence in a description is strongly dependent on the usefulness of the features used and the resulting simplicity of description. There will be later occasion to comment on this matter in detail.

4. The Arrangement of the Book

In the next chapter (Chapter II) Marino's model will be introduced formally and some of its important properties discussed. It will then be shown how some of the important concepts associated with the Marino model can be specialized to the Windeknecht model. Some important classes of sets associated with solution methods will be isolated and discussed. It will be pointed out how some of these classes have already been used in some case studies reported in literature.

In Chapter III the Marino model will be specialized to the case of two-person games and a discussion similar to that in Chapter II will be instituted.

As a prelude to Chapter V on Pattern Recognition, Chapter IV will introduce in a precise and detailed manner the description language introduced in Sec. 3 above. This will enable the discussion of similarities and differences between various pattern recognition schemes.

In all these Chapters certain statements made in this present Chapter will be established precisely.

In Chapter VI the role of pattern recognition in problem and game solving will be discussed. The importance of the appropriateness of the description language will be brought out in view of the crucial role it plays in learning solution methods. Some examples will be given. These will be simple, merely because very few difficult examples have been worked out.

In Chapter VII short descriptions will be given of certain research activities at the Case Institute of Technology where computer programs have been implemented for certain problem solving tasks. The success and failure will be discussed in view of the theories discussed previously.

CHAPTER II - PROBLEMS AND SOLUTION METHODS

1. Introduction

The main purpose of this chapter is to discuss problems as modelled by Windeknecht (these will be called W-problems in the future). However, since many ideas relevant to Marino's model is relevant to this as well as to the next Chapter, the next two sections of the present Chapter will be devoted to Marino's model (hereinafter called M-situations), to some of its properties and to its relationship with W-problems. An important theorem regarding M-situations (Theorem 2.1 below) deals with the existence of winning strategies in M-situations. Similar theorems will be shown to exist for W-problems and Game-Situations (discussed in Chapter III). These will be established by establishing W-problems and Game-Situations as special cases of M-situations. To enable one to do this it is necessary to i) set down the basic structure of M-situations, ii) to put down the basic structure of W-problems and indicate their isomorphism with a special class of M-situations and iii) to set down the basic structure of Game-Situations and indicate their isomorphism with another special class of M-situations. Sec. 2 will formalize the structure of M-situations. Section 3 will elaborate the discussions envisaged in (ii) above. The discussion of relationships between M-situations and Game Situations will be relegated to Chapter III.

In Sections 4 and 5 some well known problems and puzzles will be described as W-problems. In later sections, some methods for finding solutions to problems will be discussed. It will be shown to what extent these methods have been approximated by some solution methods used in literature.

2. Some Properties of M-Situations

In Chapter I, the basic ideas underlying M-situations have been clarified. In what follows some of these ideas will be made more precise. As said before, this formalism is essentially those of Marino, although a few minor changes have been made to bring them in line with the purposes of this book.

An M-situation is given by a 7-tuple $< S,C,D,M,S_W,S_L >$ where S, C and D are abstract sets and S_W and S_L are disjoint subsets of S. M is a subset of SxCxDxS with the following properties.

M1. $(s_1,c_1,d_1,s_2) \in M$ and $(s_1,c_1,d_1,s_3) \in M$ implies $s_2 = s_3$.

This merely says that M is a function mapping a subset of SxCxD into S. The reason it is not defined initially as a function is because M is not defined for the entire set SxCxD.

Before the next properties of the relation M are introduced another definition will be needed. Given an M-situation and an element $c \in C$ one defines the set

$$S_c = \{s \mid (\exists d)(\exists s')((s,c,d,s') \in M)\} .$$

Similarly, for each member $d \in D$ one can define

$$S_d = \{s \mid (\exists c)(\exists s')((s,c,d,s') \in M)\} .$$

It follows from the definition that if $(s,d,c,s') \in M$, then $s \in S_c \cap S_d$. However, in all M-situations it will further be assumed that

M2. If $s \in S_c \cap S_d$ then $(\exists s')((s,c,d,s') \in M)$

For convenience as well as for motivation, members of S will be called situations; members of S_W and S_L will be called winning situations

and losing situations respectively; members of C will be called controls and members of D will be called disturbances. $(s, c,d,s') \in M$ will often be expressed by saying "s' is the result of applying c and d to s" or by $(c,d)(s) = s'$. Members of S_c will be called "situations to which c is applicable". Similarly with S_d. Situations to which no controls are applicable, if not winning or losing situations, will be called draw situations and denoted by S_D.

A function

$$P: S-(S_W \cup S_L \cup S_D) \rightarrow C$$

will be called a control strategy if

$$P(s) = c \text{ implies } s \in S_c .$$

A disturbance strategy is defined similarly

A winning strategy is one such that, no matter what strategy is chosen by the disturbing influence, any sequence of applications of controls and disturbances applied according to the strategies results in a winning situation. One can express this formally as follows.

Given an element $s_o \notin S_W \cup S_L \cup S_D$, a control strategy P_C is called a winning strategy for s_o if for every disturbance strategy P_D there exists a sequence $(c_1,d_1),(c_2,d_2),(c_3,d_3),\ldots(c_n,d_n)$ such that

$$c_1 = P_C(s_o),\ d_1 = P_D(s_o);$$

and for each i $(1 \leq i < n)$:

$$c_{i+1} = P_C((c_i,d_i)((c_{i-1},d_{i-1})(\ldots(c_1,d_1)(s_o)\ldots),$$

$$d_{i+1} = P_D((c_i,d_i)((c_{i-1},d_{i-1})(\ldots(c_1,d_1)(s_o)\ldots);$$

and

$$(c_n,d_n)((c_{n-1},d_{n-1})(\ldots(c_1,d_1)(s_o)\ldots) \in S_W.$$

A control strategy P_C is called a non-losing strategy for s_o if it is either a winning strategy or for no disturbance strategy P_D is it the case that there exists a sequence $\{(c_i,d_i)\}$ $(1 \leq i \leq n$ as before) such that

$$c_1 = P_C(s_o),\ d_1 = P_D(s_o);$$

and for each i $(1 \leq i < n)$

$$c_{i+1} = P_C((c_i,d_i)((c_{i-1},d_{i-1})(\ldots(c_1,d_1)(s_o)\ldots),$$

$$d_{i+1} = P_D((c_i,d_i)((c_{i-1},d_{i-1})(\ldots(c_1,d_1)(s_o)\ldots);$$

and

$$(c_n d_n)((c_{n-1},d_{n-1})(\ldots(c_1 d_1)(s_o)\ldots) \in S_L.$$

A situation for which a winning strategy exists is called a forcing situation. The set of all forcing situations is denoted by S_F. A situation for which a non-losing strategy exists but no winning strategy exists is called a neutral situation.

The following theorem is of great interest. We shall state it here without proof (a proof can be found in Marino's thesis).

Theorem 2.1. Given an M-situation, there exists a strategy which is a winning strategy for every forcing situation and a non-losing strategy for every neutral situation.

In the next section the definition of W-problems will be introduced and related to that of M-situations. It may be worthwhile to mention at this point, however, that the word "strategy" here has been used in a somewhat specialized sense. Unlike its traditional usage in the field, a strategy is not a method for searching for the solution. In a later section this latter method is called a "search strategy". A strategy, as the

term is used here, is the embodiment of the construction of a solution - correctly or otherwise. A winning strategy is a method for embodying the construction of a "correct" solution. A search is really a method for changing strategies as defined here.

3. W-Problems and M-Situations

A W-problem is given by a triple $\langle S, F_o T\rangle$ where S is an abstract set, T a subset of S and F_o a set of functions from subsets of S into S

$$f \in F_o \text{ implies } f : S_f \to S \text{ and } S_f \subseteq S.$$

Given a W-problem and $s_o \in S$, a winning solution for s_o is a sequence of functions $f_1, f_2, \ldots f_n$ such that $f_i \in F_o$ for each i and such that

$$f_n(f_{n-1}(\ldots f_1(s_o)\ldots) \in T .$$

A function

$$Q : \bigcup_{f \in F_o} S_f - T \to F_o$$

is called a W-strategy if and only if

$$Q(s) = f \text{ implies } s \in S_f .$$

A W-strategy is winning for $s_o \in S$ if there exists a winning solution $f_1, f_2, \ldots f_n$ such that

$$f_1 = Q(s_o)$$

and for each i $(1 \leq i \leq n)$

$$f_{i+1} = Q(f_i(f_{i-1}(\ldots f_1(s_o)\ldots).$$

To indicate the relationship between W-problems and M-situations one needs a special class of M-situations which will be defined next.

An M-situation is called problem-like if and only if

P1. $D = \{d^r\}$, a unit set.

P2. $s \in S_c$ for some c implies $s \notin S_L$.

P3. $S_D = \emptyset$ the empty set.

Given a problem-like M-situation,

$$R = \langle S, C, (d'), M, S_W, S_L \rangle$$

one can define a triplet

$$P(R) = \langle S, F_o, T \rangle$$

where $T = S_W$ and F_o is a set of relations defined as follows:

$f \in F_o$ if and only if there exists a $c \in C$ such that for all s, $s' \in S$

$(s,s') \in f$ if and only if $(s,c,d',s') \in M$.

It is not hard to see that $P(R)$ is a W-problem. S is an abstract set, and T is a subset of S. Each element $f \in F_o$ is a function since $(s,s') \in f$ and $(s,s'') \in f$ implies $(s,c,d',s') \in M$ and $(s,c,d',s'') \in M$ for some unique c whence by M1, $s' = s''$. For each $c \in C$ there is a function $f_c \in F_o$ and these are the only members of F_o. Also $S_{f_c} = S_c$. This can be seen by noting that $s \in S_c$ implies (due to the uniqueness of d') $s \in S_c \cap S_{d'}$, whence by definition there exists s' such that $(s,c,d',s') \in M$ whence f_c is defined for s; proving $S_c \subseteq S_{f_c}$. Similarly, if $s \in S_{f_c}$, then there is an s' such that $(s,c,d',s') \in M$ whence $s \in S_c$.

Given a W-problem

$$P = \langle S, F_o, T \rangle$$

one can define a 7-tuple

$$R(P) = \langle S, C, (d'), M, S_W, S_L \rangle$$

where

$$S_W = T,$$

$$S_L = S - \bigcup_{f \in F_o} S_f - S_W$$

and C and M is composed as follows:

For each element of $f \in F_o$ there is an element $c_f \in C$ and these are the only elements of C. Also

$$(s, c_f, d', s') \in M \text{ if and only if } f(s) = s'.$$

R(P) is a problem-like M-situation since S and C can be taken to be abstract sets, $D = \{d'\}$ satisfying P1. S_W and S_L are disjoint subsets of S by definition. M1 follows since f is a function for each c_f. M2 follows also since from the uniqueness of d', $s \in S_{c_f}$ implies and (trivially) is implied by $s \in S_{c_f} \cap S_{d'}$. Since it can also be seen quite easily that $S_f = S_{c_f}$, $s \in S_{c_f}$ also implies the existence of s' such that $f(s) = s'$ leading to $(s, c_f, d', s') \in M$. P2 can be seen to be satisfied since $s \in S_L$ shows $s \in S - \bigcup_{c \in C} S_c$, i.e. $s \notin \bigcup_{c \in C} S_c$ proving the contrapositive of P2. To prove P3 one notices that

$$S_D = S - \bigcup_{c \in C} S_c - S_L - S_W \text{ and } S_L = S - \bigcup_{c \in C} S_c - S_W.$$

In what has gone above, two mappings have been defined, one from M-situations to W-problems and one from W-problems to M-situations. In what follows it is shown that these mappings are one-one and inverses of one another.

<u>Theorem 2.2.</u> For all W-problems P and problem-like M-situations R

$$P(R(P)) = P \quad \text{and} \quad R(P(R)) = R.$$

<u>Proof</u>:

$$\text{Let } P = \langle S, F_o, T \rangle$$

$$R(P) = \langle S, C, \{d'\}, M, T, S - \bigcup_{c \in C} S_c - T \rangle$$

$$\text{and } P(R(P)) = \langle S, F_o', T \rangle$$

It is only required to show that $F_o = F_o'$

Let $f \in F_o$; then by definition of R(P), there exists a $c_f \in C$ such that $S_{c_f} = S_f$ and $f(s) = s'$ if and only if $(s, c_f, d', s') \in M$. By construction of F_o', there is an element $f' \in F_o'$ such that $S_{f'} = S_{c_f}$ and

$f'(s) = s'$ if and only if $(s, c_f, d', s') \in M$. Hence $f' = f$ showing that $F_o \subseteq F_o'$. The reverse inequality follows similarly proving that $F_o = F_o'$.

Let now

$$R = \langle S, C, \{d'\}, M, S_W, S_L \rangle$$

$$P(R) = \langle S, F_o, S_W \rangle$$

$$\text{and } R(P(R)) = \langle S, C', \{d'\}, M', S_W, S_L' \rangle$$

That $C = C'$ and $M = M'$ will follow from definition as in the previous case. To show that $S_L = S_L'$, it is recalled that from P2 that $s \in S_L$ implies $s \notin \bigcup_{c \in C} S_c$ or $S_L \subseteq S - \bigcup_{c \in C} S_c$. However, since S_L are S_W are disjoint,

$$S_L = S_L - S_W \subseteq S - \bigcup_{c \in C} S_c - S_W = S_L' .$$

Again, since by P3

$$S_D = S - \bigcup_{c \in C} S_c - S_L - S_W = \phi$$

one has

$$S_L \supseteq S - \bigcup_{c \in C} S_c - S_W = S_L'$$

proving the reverse inequality.

The R and P functions demonstrate that problem-like M-situations and W-problems are identical structures. However, they do not establish that the concept of a winning strategy as defined in the two structures are identical concepts. To show this one introduces another definition.

Given a W-problem P and a W-strategy Q for P, one defines a function R(Q) from a subset of S into C as follows

$$R(Q)(s) = c_f \text{ if and only if } Q(s) = f.$$

It can be verified that R(Q) is a control strategy for R(P). $R(Q)(s) = c_f$ implies $Q(s) = f$. Since Q is a W-strategy, this implies $s \in S_f$ whence $s \in S_{c_f}$ showing that R(Q) fulfills one condition for being a

control strategy. To show that the domain of $R(Q)$ is indeed $S-S_W-S_L-S_D$, one notices that the domain of Q is $\bigcup_{f\in F_o} S_f - S_W = \bigcup_{c\in C} S_c - S_W$. Now

$$S_L = S - \bigcup_{c\in C} S_c - S_W$$

whence

$$S-S_L = \bigcup_{c\in C} S_c \cup S_W$$

whence

$$S-S_L-S_W = \bigcup_{c\in C} S_c - S_W$$

Since $S_D = \phi$ the domain of $R(Q)$, which coincides with the domain of Q is indeed $S-S_W-S_L-S_D$.

One can now state and prove the following theorem, which is an important step towards the establishment of the analog of Theorem 2.1, in the case of W-problems.

<u>Theorem 2.3.</u> Let Q be a W-strategy for the W-problem P. Let $R(Q)$ be the control strategy for $R(P)$ as defined. ThenQ is a winning W-strategy for s_o if and only if $R(Q)$ is a winning control strategy for s_o in $R(P)$.

<u>Proof:</u>

One initially notices from the construction of M that $s\in S'_d$ if and only if $s\in S_f$ for some $f\in F_o$. Hence every disturbance strategy P_D has as its domain $\bigcup_{f\in F_o} S_f - S_W$ and $P_D(s) = d'$ for each s in this domain. This, then, is the only possible disturbance strategy.

Let now $R(Q)$ be a winning strategy for s_o in $R(P)$. Then there exists a sequence of controls $c_{f_1}, s_{f_2}, \ldots c_{f_n}$ such that

$$c_{f_1} = R(Q)(s_o)$$

and for each $i < n$

$$c_{f_i} = R(Q)((c_{f_{i-1}},d')((\ldots(c_{f_1},d')(s_o))\ldots)$$

and

$$(c_{f_n},d')(c_{f_{n-1}},d')((\ldots((c_{f_i},d')(s_o)\ldots) \in S_W$$

This indicates that

$$f_n(f_{n-1}(\ldots f_1(s_o)\ldots) \in S_W$$

$$f_1 = Q(s_o)$$

and

$$f_i = Q(f_{i-1}(\ldots f_1(s_o)\ldots) \qquad \text{for each } i \leq n.$$

Thus Q is a winning W-strategy for s_o in P. The proof that R(Q) is a winning strategy for s_o in R(P) if Q is a winning W-strategy for s_o in P follows in exactly the same way. It can also be shown that any strategy in R(P) is R(Q) for some strategy Q in P.

One can state without proving that if there is a winning solution for s_o in P then there is a winning W-strategy for s_o in P. One merely associates f_1 with s_o, f_2 with $f_1(s_o)$ and so on; the rest of the situations have any value for the strategy.

Let $T' \subseteq S$ be the set of all situations in a W-problem such that $s \in T'$ if and only if there is a winning solution for s. Hence there is a winning strategy for each element of T' in R(P). Hence each element of T' is a forcing state in R(P). Also, each forcing state in R(P) is a member of T'.

The important thing to note here is that by theorem 2.1 there is a strategy R(Q) in R(P) which is a winning strategy for each $s \in T'$. Hence there is a W-strategy which is winning for each element $s \in T'$. This fact yields some solidity to a rather meaningful theorem that will be indicated

later. In the rest of this chapter some processes for finding winning strategies will be discussed. To make this discussion meaningful, the next two sections will introduce two problems that have been discussed in literature and show that these can be formalized as W-problems.

4. A Simple Example of a W-Problem: The Tower of Hanoi.

In the present section and the next, two problems will be discussed as W-problems. In addition to motivating the use of W-problems, these would also serve in future sections to illustrate some ideas developed in relation to solution methods. There will also be occasion to indicate how some of these ideas are inherent in methods described in Artificial Intelligence literature.

This present section will be devoted to describing the celebrated puzzle called the tower of Hanoi [11 , 12]. The puzzle is generally described as follows.

One is given a set of n disks (n may be any number: folklore attaches the value 64 to n; we shall exemplify our problem with smaller values). These are of unequal diameter. There are three long pins fixed upright on a board. Each disc has a hole in the center large enough to pass any pin through it but not large enough to pass any other disc through it.

Initially, all the discs are on one of the pins. They are arranged with the largest disc at the bottom and the smallest disc at the top; each disc rests on a larger disc. It is required to transfer all the discs to another pin by moving one disk at a time from one pin to another. The constraint is to be observed at all times that no disc should ever rest on any disc smaller than itself. Also only the discs at the top position in any needle can be moved. The initial configuration is shown in Figure 2.1, to clarify the description given. n has been taken to be 6 here. However, the value of n will not play any essential role in the formal representation of the problem.

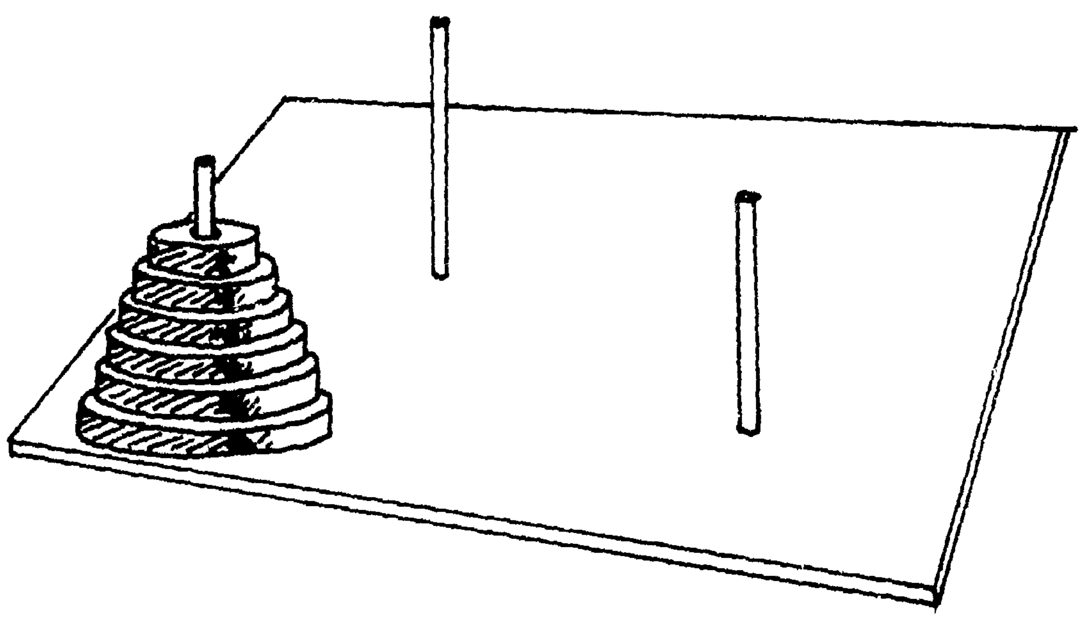


Figure 2.1. The Tower of Hanoi

To represent the Tower of Hanoi problem as a W-problem one may specify as follows.

Each element of S consists of a sequence of three sequences of integers

$$s \in S \equiv s = <(x_{01}, x_{02}, \ldots x_{0i_0}), (x_{11}, x_{12}, \ldots x_{1i_1}), (x_{21}, x_{22}, \ldots x_{2i_2})>$$

These three sequences have the following properties:

H1. $i_0 + i_1 + i_2 = n$

(i.e., there are a totality of n integers in the three sequences, each integer standing for a disc).

H2. $x_{ij} = x_{st}$ only if $i = s$, $j = t$

(i.e., the integers appearing in the sequences are distinct)

H3. $1 \leq x_{ij} \leq n$

H4. For each i, $j > k$ implies $x_{ij} > x_{ik}$

(i.e., larger integers appear after smaller ones in each sequence: smaller integers stand for larger discs)

The set F_o consists of 2n functions, to be denoted by the generic name (k,m) where $1 \leq k \leq n$ and $m = +1$ or -1. The move (k,m) is the formal analog of moving the $k^{\underline{th}}$ disc (in the order of size) from the top of any pile (or pin) either to the pile to the left or the one to the right depending on the sign of m. The domain $S_{(k,m)}$ of the function (k,m) is defined as follows:

H5. $<(x_{01}, x_{02}, \ldots x_{0i_o}), (x_{11}, x_{12}, \ldots x_{1i_1}), (x_{21}, x_{22}, \ldots x_{2i_2})>$

is an element of $S_{(k,m)}$ if and only if either $x_{0i_0} = k$ or $x_{1i_1} = k$ or $x_{2i_2} = k$ and if $x_{si_s} = k$ then $k > x_{ti_t}$ where $t = s + m \pmod 3$. The values of the functions are given as follows.

H6. If $s = \langle(x_{01}, x \quad, \ldots x_{0i_0}), (x_{11}, x_{12}, \ldots x_{1i_1}), (x_{21}, x_{22}, \ldots x_{2i_2})$

is in the domain of (n) and $k = x_{ji_j}$ $(j = 0,1,2)$ then

$(k,m)(s) = \langle(x'_{01}, \quad _2, \ldots x'_{0i_0}), (x'_{11}, x'_{12}, \ldots x'_{1i_1}), (x'_{21}, x'_{22}, \ldots x'_{2i_2})\rangle$

where

$i'_t = i'_t - 1$ if $t = j$; $i'_t = i_t + 1$ if $t = j + m \pmod 3$; $i'_t = i_t$ otherwise.

Also

$$x'_{ti} = x_{ti} \text{ except when } t = j + m \pmod 3 \text{ and } i = i'_t$$

$$x'_{ti'_t} = k \text{ when } t = j + m \pmod 3.$$

H7. T is specified to be the unit set consisting of $\langle \phi, (1,2,\ldots n), \phi \rangle$.

Interest will be centered on specifying a winning solution for $\langle (1,2,\ldots n), \phi, \phi \rangle$.

As an example of a solution of the problem when n = 3, let us consider the following sequence of moves. The smallest disc is moved to the right pin, the second largest disc is moved to the left pin; the smallest disc is then moved right (from the right to the left pin "around the circle") to the top of the second largest pin. Then the largest disc is moved to the (now empty) right pin, the smallest disc moved right to the original (now empty) pin, the second largest disc is moved left (around the circle) to the top of the largest disc on the right pin and the smallest disc moved right to the top of the pile.

The exhibition of this sequence of moves is formally equivalent to the statement that when n = 3

$$(3,+1),(2,-1),(3,+1),(1,+1),(3,+1),(2,-1),(3,+1)$$

is a solution, since

(3,+1)((2,-1)((3,+1)((1,+1)(3,+1)((2,-1)((3,+1)(<(1,2,3), ϕ, ϕ >)))))))

(3,+1)((2,-1)((3,+1)((1,+1)((3,+1)((2,-1)(<(1,2,),(3), ϕ >))))))

(3,+1)((2,-1)((3,+1)((1,+1)((3,+1)(<(1),(3),(2)>)))))

(3,+1)((2,-1)((3,+1)((1,+1)(<(1), ϕ, (2,3)>))))

(3,+1)((2, 1)((3,+1)(< ϕ, (1),(2,3)>)))

(3,+1)((2,-1)(<(3),(1),(2)>))

(3,+1)(<(3),(1,2), ϕ >)

< ϕ, (1,2,3), ϕ >.

In a later section a winning strategy will be pointed out which yields this winning solution. Meanwhile, the next section will formalize the problem of finding proofs in Propositional Calculus. For this purpose the Russel-Whitehead version of the propositional calculus will be used, following Simon, Newell and Shaw.

5. The Logic Theorist - Another Example

As an introduction to formalizing the problem of finding proofs in propositional calculus as a W-problem, the exact model used in the logic theorist will be discussed here. This example will indicate the motivation for considering W-problems to have a possibly infinite F_o, as has been done in this book.

As in the literature, the alphabet of the propositional calculus will consist of the following:

1. An infinite set of propositional variables, whose members will be denoted by lower case latin letters with integral subscripts if necessary.

2. The symbols (,), $\sim$, $\vee$ and $\rightarrow$.

Well-formed formulae (wffs) are defined in the usual way as follows:

a) Any propositional variable is a wff.

b) If A and B are wffs, then $\sim A$, $(A \vee B)$, $(A \rightarrow B)$ are wffs.

It is understood that A and B above are meta-linguistic variables standing for wffs. Also, in keeping with literature, parentheses may be dropped when exhibiting a wff, it being understood that the resulting strings of characters are shorthands for wffs rather than wffs themselves.

A subset of the set of wffs is defined as the set of theorems as follows:

0. The following (called axioms) are theorems

(i) $((p \vee p) \rightarrow p)$

(ii) $(p \rightarrow (q \vee p))$

(iii) $((p \vee q) \rightarrow (q \vee p))$

(iv) $((p \vee (q \vee r)) \rightarrow (q \vee (p \vee r)))$

(v) $((p \rightarrow q) \rightarrow ((r \vee p) \rightarrow (r \vee q)))$

1. If A(p) is a theorem in which the specific propositional variable p occurs, and B is any wff, then A(B) is a theorem, where A(B) is obtained from A(p) by replacing every occurence of p in A(p) by an occurence of B. This is the usual substitution rule.

2a. If ($\sim A \vee B$) occurs anywhere in a theorem C, then D, obtained by replacing this occurence in C by ($A \rightarrow B$), is also a theorem.

2b. If ($A \rightarrow B$) occurs in any theorem C, then D, obtained by replacing this occurence in C by ($\sim A \vee B$), is also a theorem. 2a and 2b are applications of the definition of "implication" in terms of "not" and "or".

3. If ($A \rightarrow B$) is a theorem and A is a theorem, then B is a theorem. (The usual modus ponens.)

On the basis of these definitions, one can set up a W-problem as follows to represent the problems of finding the proofs of theorems.

Each situation s is a finite sequence of wffs $(s_1, s_2, \ldots s_n)$. These wffs stand for the set of theorems proved at a certain stage of the proof procedure.

F_o consists of four classes of functions, denoted by (i) (i, A, B,1) where i is an integer, A is a propositional variable and B is a wff, (ii) (i,j,2a) where i and j are integers (iii) (i,j,2b) where i and j again are integers and (iv) (i,j,3) where i and j are integers. These stand for substitution, tne "forward" and "backward" application of the definition of implication and modus ponens respectively. Formally, these functions are defined as follows:

(i) (Substitution) $s = (s_1, s_2, \ldots s_n)$ is a member of $S_{(i,A,B,1)}$ if and only if $i \leq n$ and the wff s_i contains the propositional variable A. In this case

$$(i,A,B,1)((s_1,s_2,\ldots s_n)) = (s_1,s_2,\ldots s_n,s_{n+1})$$

where s_{n+1} is obtained from s_i by replacing all occurences of A by B.

(ii) (Definition-Application) $s = (s_1,s_2,\ldots s_n)$ is a member of $S_{(i,j,2a)}$ if and only if $i \leq n$ and there are at least j occurences of wffs of the form $(\sim A \vee B)$ in s_i. In this case

$$(i,j,2a)((s_1,s_2,\ldots s_n) = (s_1,s_2,\ldots s_n,s_{n+1})$$

where s_{n+1} is obtained by replacing the jth occurence of $(\sim A \vee B)$ in s_i by $(A \rightarrow B)$.

(iii) (Definition-Application) $s = (s_1,s_2,\ldots s_n)$ is a member of $S_{(i,j,2b)}$ if and only if $i \leq n$ and there are at least j occurences of wffs of the form $(A \rightarrow B)$ in s_i. In this case

$$(i,j,2b)((s_1,s_2,\ldots s_n) = (s_1,s_2,\ldots s_n,s_{n+1})$$

where s_{n+1} is obtained by replacing the jth occurence of $(A \rightarrow B)$ in s_i by $(\sim A \vee B)$.

(iv) (Modus Ponen) $s = (s_1,s_2,\ldots s_n)$ is a member of $S_{(i,j,3)}$ if and only if $i,j \leq n$ and for some wffs A and B, s_i is $(A \rightarrow B)$ and s_j is A. In this case

$$(i,j,3)((s_1,s_2,\ldots s_n) = (s_1,s_2,\ldots s_n,s_{n+1})$$

where s_{n+1} is B.

In (ii) and (iii) above the occurences of the sentence of the form $(\sim A \vee B)$ and $(A \rightarrow B)$ are ordered by the occurences of the main connectives of these sentences, reading from left to right. As an example, in the wff

$$C = ((\sim(\sim a \vee (a \rightarrow b)) \vee (c \rightarrow (\sim d \vee e))) \rightarrow f)$$

the first occurence of a wff of the form $(\sim A \vee B)$ is $(\sim a \vee (a \to b))$; the second occurence is $(\sim(\sim a \vee (a \to b)) \vee (c \to (\sim d \vee e)))$; the third occurence is $(\sim d \vee e)$. Similarly the first occurence of a sentence of the form $(A \to B)$ is $(a \to b)$; the second occurence is $(c \to (\sim d \vee e))$; the third occurence is C itself. Hence if the situation s consists of the single element C, then

$$(1,2,2a)(s) = (((\sim(\sim a \vee (a \to b)) \vee (c \to (\sim d \vee c))) \to f),(((\sim a \vee (a \to b)) \to (c \to (\sim d \vee e))) \to f))$$

while

$$(1,3,2b)(s) = (((\sim(\sim a \vee (a \to b)) \vee (c \to (\sim d \vee e))) \to f), (\sim(\sim(\sim a \vee (a \to b)) \vee (c \to (\sim d \vee e))) \vee f)).$$

Finding a proof of B from the suppositions $A_1,A_2,\ldots A_k$ (i.e., showing $A_1,A_2,\ldots A_k \vdash B$) would correspond to the following W-problem. s_o is the sequence $(A_1,A_2,\ldots A_k,X_1,X_2,\ldots X_5)$ where $X_1,X_2,\ldots X_5$ are the five axioms. T consists of the set of all sequences of wffs containing B. A winning solution would be a sequence $f_1,f_2,\ldots f_n$ of functions such that the sequence

$$f_n(f_{n-1}(\ldots f_1((A_1,A_2,\ldots A_k,X_1,X_2,\ldots X_5))\ldots)\epsilon\ T$$

As an example, the winning solution of the W-problem corresponding to a proof of $((p \to \sim p) \to \sim p)$ from the axioms $(\vdash((p \to \sim p) \to \sim p))$ would be the sequence

$$(1,p,\sim p,1),(6,1,2a)$$

This can be seen as follows

$$(6,1,2a)((1,p,\sim p,1)((((p \vee p) \to p),(p \to (q \vee p)), ((p \vee q) \to (q \vee p)), ((p \vee (q \vee r)) \to (q \vee (p \vee r))),((p \to q) \to ((r \vee p) \to (r \vee q))))))$$

$$= (6,1,2a)((((p \vee p) \rightarrow p),(p \rightarrow (q \vee p)),((p \vee q) \rightarrow (q \vee p)),((p \vee (q \vee r)) \rightarrow (q \vee (p \vee r))),((p \rightarrow q) \rightarrow ((r \vee p) \rightarrow (r \vee q))),((\sim p \vee \sim p) \rightarrow \sim p)))$$

$$= (((p \vee p) \rightarrow p),(p \rightarrow (q \vee p)),((p \vee q) \rightarrow (q \vee p)),((p \vee (q \vee r)) \rightarrow (q \vee (p \vee r))), ((p \rightarrow q) \rightarrow ((r \vee p) \rightarrow (r \vee q))),((\sim p \vee \sim p) \rightarrow \sim p),((p \rightarrow \sim p) \rightarrow \sim p)).$$

The last sequence contains the theorem to be proved as its last element.

Two examples have been given, in this section and the previous one, of the representation of two problems as W-problems. In the next few sections various properties of winning strategies will be discussed.

6. Strategies and their Description

It is clear from the discussion in Section 3 that the solution for a W-problem can be found if a winning W-strategy is known. The idea of a strategy has been inherent in many works in Artificial Intelligence. However, the mere giving of a precise form to this idea does not shed any light on the basic question, "How is a winning strategy to be found?" In later sections various devices will be suggested for the finding of strategies. Of course, these devices in their turn will need the knowledge of other functions or sets. These again, will have to be "found" for any given problem one is faced with. These are introduced in the hope that some of these will prove easier to deduce from the description of the problem or from "experience".

In this section attention will be given to a different, no less crucial problem. "Even when one knows a strategy, how can one make sure that it is easy to implement?" That is, in what form is a strategy to be represented in memory? Evidently, the strategy cannot be stored as a huge set of ordered pairs. It is essential that a small set of tests be specified to the computer. The value of the strategy for a situation (the control to be chosen) is determined on the basis of these tests. As can be seen from our discussion in Section 3 of Chapter I, this is essentially a problem of description of sets. The details of this latter subject can only be discussed in a later chapter. For the present it will be assumed only that some subsets of S are "easier" to describe than others. It will also be assumed on the basis of the discussion in Chapter I that it is of use to be able to find a common description for all situation in S which yields the same value of the strategy.

Given a strategy Q one can define the following relation E on

$\bigcup_{f \in F_0} S_f - T$

$$(s_1, s_2) \in E \text{ if and only if } Q(s_1) = Q(s_2)$$

Clearly, this relation is an equivalence relation. This relation is precisely the relation $Q \circ Q^{-1}$ (the composition of Q with its inverse relation) and hence will be called the Kernel of the strategy Q, following algebraic terminology [13].

The Kernel of a strategy partitions the set $\bigcup_{f \in F_o} S_f - T$ into disjoint subsets called its equivalence classes. For a strategy to be practicable, each subset in this partition is to be easily describable. For future purposes, the symbol $Q^{-1}(f)$ will be used to denote the set of all points s such that $Q(s) = f$. This, again, is standard algebraic notation. It follows immediately from definitions that

$$f_1 \neq f_2 \text{ implies } Q^{-1}(f_1) \cap Q^{-1}(f_2) = \emptyset$$

and

$$\bigcup_{f \in F_o} Q^{-1}(f) = \bigcup_{f \in F_o} S_f - T$$

A particularly easy description for these sets exist for the Tower of Hanoi problem. This will now be set down in the way of an example.

It will be recalled that in the Tower of Hanoi Problem of Section 3, a situation consists of a sequence $\langle s_1, s_2, s_3 \rangle$ where each s_i $(i = 1,2,3)$ is a sequence of integers with certain properties. It will also be recalled that the controls are denoted by ordered pairs (k,ℓ) where k is an integer and $\ell = \pm 1$. $S_{(k,\ell)}$ the domain of (k,ℓ) have been previously defined. Also, it has been pointed out that there are n elements in the union of s_1, s_2, s_3, considered as sets. One now defines a strategy as follows:

$P(s) = (k,(-1)^{k+1})$ if and only if

i) $s \neq < \emptyset,(1,2,3,\ldots,n),\emptyset >$ and

ii) $s \in S_{(k,(-1)^{k+1})}$ and

iii) $s \notin S_{(i,(-1)^{k+1})}$ for any $i < k$.

Since in the case of the Tower of Hanoi $\bigcup_{f \in F} S_f = S$, the domain of P ought to be S-T which it is, by condition (i) above. Condition (ii) assumes us that the strategy always chooses an applicable move. That P is indeed a winning strategy for $s_o = <(1,2,3,\ldots,n),\emptyset,\emptyset >$ can be verified in the case of n = 3, as follows:

$P(<(1,2,3),\emptyset,\emptyset >) = (3,+1)$

and $(3,+1)(<(1,2,3),\emptyset,\emptyset>) = <(1,2),(3),\emptyset>$

$P(<(1,2),(3),\emptyset>) = (2,-1)$

and $(2,-1)(<(1,2),(3),\emptyset>) = <(1), (3), (2)>$

$P(<(1),(3),(2)>) = (3,+1)$

and $(3,+1)(<(1), (3), (2)>) = <(1), \emptyset, (2,3)>$

$P(<(1), \emptyset, (2,3)>) = (1,+1)$

and $(1,+1)(<(1), \emptyset, (2,3)>) = <\emptyset, (1), (2,3)>$

$P(<\emptyset, (1), (2,3)>) = (3,+1)$

and $(3,+1)(<\emptyset, (1), (2,3)>) = <(3), (1), (2)>$

$P(<(3), (1), (2)>) = (2,-1)$

and $(2,-1)(<(3), (1), (2)>) = <(3), (1,2), \emptyset >$

$P(<(3), (1,2),\emptyset >) = (3,+1)$

and $(3,+1)(<(3), (1,2), \emptyset >) = < \emptyset,(1,2,3),\emptyset >$

The last situation is a winning situation. It will be noticed that the strategy P has yielded the same solution as was exemplified in Section 3.

It can be shown by induction that P is indeed the winning strategy for (<(1,2,...n), Ø,Ø >) for all n. This will be indicated when the concept of subgoals is discussed later. It may be pointed out that P is the winning strategy for all states. However, for some situations it may not yield the "shortest" solution in the sense that for these situations one can find solutions as a shorter sequence of controls by violating the strategy P. There are, again, other strategies for the Tower of Hanoi which are winning strategies only for some situations. There are others which yield the shortest solution for all situations. At present, it is not proposed to discuss these various strategies. It is, however, worthwhile pointing out at this point that the strategy P discussed here has been defined mostly in terms of statements which are needed for describing the problem itself. Only the concept of taking the powers of -1 was not a part of the concepts used in the description of the rules of the game. The others - the descriptions of $S_{(k,\ell)}$ and the concept of one integer being less than another - were inherent in the description of the problem. Free use was also made of logical quantification in defining the strategy but these were used also in the description of the problem. The significance of these facts will be discussed when the basis for description languages and their use in problem solving has been made clearer.

The present section will be closed by pointing out an important consideration regarding the search for a practicable strategy. In many real problems one needs winning solutions, not for all possible situations, but only for a few situations. By the Marino theorem and Theorem 2.3, there is a winning strategy for the set of all situations for which a winning solution exists. However, it may be more advantageous to find a less "ambitious" strategy; one which is a winning strategy -- not for all possible

situations but only for those situations for which a solution is needed. This will be clarified through a theorem. To introduce the theorem one needs the following definition.

Given any subset $S_1 \subseteq T'$ let $\mathcal{P}(S_1)$ denote the set of all strategies which are winning strategies for every element of S_1, that is $P \in \mathcal{P}(S_1)$ if and only if for every $s \in S_1$, P is a winning strategy for s.

<u>Theorem 2.4</u> If $S_1 \subseteq S_2 \subseteq T'$ then $\mathcal{P}(S_1) \supseteq \mathcal{P}(S_2) \neq \emptyset$.

<u>Proof:</u>

Let $P \in \mathcal{P}(S_2)$. Then P is a winning strategy for all elements of S_2. Since each element of S_1 is an element of S_2, P is a winning strategy for every element of S_1. That is, $P \in \mathcal{P}(S_1)$.

That $\mathcal{P}(S_2) \neq \emptyset$ follows from Theorem 2.1 which states that $P(T') \neq \emptyset$ and that $P(S_2) \supseteq P(T')$.

The theorem gives us no assurance that if S_1 is a proper subset of S_2 then $\mathcal{P}(S_2)$ is a proper subset of $\mathcal{P}(S_1)$. This is in general not true, either. However, there are many cases where choosing a proper subset of a set of situations yields a larger set of available winning strategies.

Let there be some evaluation function which associates with every subset of S a number which yields the "ease" with which it can be described. Then with each strategy P we can associate a set of numbers each corresponding to the ease with which an equivalence class of its Kernel (PoP^{-1}) can be described. The minimum of these numbers can be used as a measure of the "ease" with which P can be used. Associated with each subset S_1 of T', then, is a set of numbers, corresponding to the ease with which each strategy in $P(S_1)$ can be used. Let the minimum of these be

denoted by $E(S_1)$. Since, by the above theorem $S_1 \subsetneq S_2$ implies $\hat{\mathfrak{S}}(S_2) \subseteq \hat{\mathfrak{S}}(S_1)$, it is not hard to see that $S_1 \subsetneq S_2$ implies $E(S_1) \geq E(S_2)$. If it is required to find a winning strategy only for all elements of S_1, it never increases the ease of applying a strategy by choosing one which is a winning strategy for a set larger than S_1.

The above gives plausible arguments for restricting ones ambition to find a winning strategy for the smallest set of situations one can "get away with". The arguments certainly are not rigorous. It has been assumed that all sets of numbers have maxima. It has been assumed that "ease" can be measured by numbers or at least by a linearly ordered set. It might be interesting to investigate the effect of relaxing these assumptions on the validity of the arguments. This will not be attempted here.

The ease of describing a strategy is only one of the problems associated with the concept of strategies. There still remains the problem of finding a strategy. In the following sections it will be pointed out that a winning strategy P can be found if certain subsets of S (other than the equivalence class of PoP^{-1}) can be easily described.

Evaluations: A Method for Defining Strategies

All the previous discussions in this Chapter lead to one important conclusion so far; to wit "In a W-problem, a winning solution for s_o can be found if one knows the description of the equivalence classes of the Kernel of some winning strategy for s_o". In the rest of this chapter, some similar statements will be made and proved where the words "the equivalence classes of the Kernel of some winning strategy for s_o" will be replaced by the names of other classes of sets. A posteriori, these also yield methods for constructing winning strategies for s_o, and these constructions will be discussed. The problem, "Given the definition of a class of sets, how does one construct their description?" will not be discussed till after description languages are introduced in Chapter IV. Unfortunately, even there the discussion will have to be sketchy.

A class of sets that readily comes to mind arises from one's desire to know how "far" a situation is from the "nearest" winning situation. This can be formalized by defining an enumerable class of sets T_i as follows

$$T_o = T$$

and for all $i > 0$

$$T_{i+1} = \{s \mid s \notin \bigcup_{k=0}^{i} T_k \text{ and } (\exists f)(f \in F_o \text{ and } f(s) \in T_i)\}$$

In words, a situation is in T_{i+1} if and only if there is a control which moves the situation to one T_i and no control which moves it any "closer" to a winning situation. The "distance" of any point from the winning situation is the minimum number of steps of elementary controls which changes it to a winning situation.

It is clear from the definition that $i \neq j$ implies $T_i \cap T_j = \emptyset$. For assuming (without less of generality) that $j < i$, one can obtain from

definition $T_i \subseteq S - \bigcup_{k<i} T_k \subseteq S - T_j$ if $j < i$. One can also see that:

<u>Theorem 2.5</u> In a W-problem $s_o \in \bigcup T_k$ if and only if $s_o \in T$ or a winning solution exists for s_o.

<u>Proof:</u>

One notes initially that $s \notin \bigcup_{k=0}^{i} T_k$ and $(\exists f)(f \in F_o$ and $f(s) \in T_i) \rightarrow s \in T_{i+1}$ or

$$(\exists f)(f \in F_o \text{ and } f(s) \in T_i) \rightarrow ((s \notin \bigcup_{k=0}^{i} T_k) \rightarrow s \in T_{i+1})$$

or

$$(\exists f)(f \in F_o \text{ and } f(s) \in T_i) \rightarrow (s \in \bigcup_{k=0}^{i} T_k \text{ or } s \in T_{i+1})$$

or

$$(\exists f)(f \in F_o \text{ and } f(s) \in T_i) \rightarrow s \in \bigcup_{k=0}^{i+1} T_k$$

from which one obtains

$$(\exists f)(f \in F_o \text{ and } f(s) \in \bigcup_{k=0}^{i} T_k) \rightarrow s \in \bigcup_{k=0}^{i+1} T_k.$$

One now proves by induction the following (yielding a stronger statement then the "if" part of the theorem).

If $f_n(f_{n-1}(\ldots f_1(s_o)\ldots) \in T$ then $s_o \in \bigcup_{i=0}^{n} T_i$. For $n = 1$ one obtains $f_1(s) \in T = T_o$ yields $s \in \bigcup_{k=0} T_k$.

Let the theorem be ture for $n = i$ that is, assume $f_i(f_{i-1}(\ldots f_1(s_o)\ldots) \in T$ yields $s_o \in \bigcup_{k=0}^{i} T_k$. Let $f_{i+1}(f_i(\ldots(f_1(s_o)\ldots) \in T$ consider $f_1(s_o) = s_1$. One has

$$f_{i+1}(f_i(\ldots f_2(s_1)\ldots) \in T$$

yielding by induction hypothesis $s_1 \in \bigcup_{k=0}^{i} T_k$. Since $f_1(s_o) = s_1 \in \bigcup_{k=0}^{i} T_k$

$$s_o \in \bigcup_{k=0}^{i+1} T_k$$

proving the theorem for all n.

To prove the converse, let $s_o \in \bigcup_i T_i$. Then there exists an n such that $s_o \in T_n$. The proof will be by induction on n, as before that either n = 0, or there exists a sequence $f_1 \ldots f_n$ of function F_o such that $f_1(f_2(\ldots f_n(s_o)) \in T$. Let n = 1, then by definition there exists $f_1 \in F_o$ such that $f_1(s_o) \in T$.

Assume as induction hypothesis that if $s_i \in f_i$, then for some $f_1, \ldots f_i$

$$f_1(f_2(\ldots f_i(s_o)\ldots) \in T.$$

Let $s_o \in T_{i+1}$. Let $f_{i+1}(s_o) \in T_i$ hence there exists a sequence of functions $f_1 \ldots f_i$ such that

$$f_1(f_2(\ldots f_i(f_{i+1}(s_o)\ldots) \in T$$

proving the theorem for all n.

The class of sets $\{T_i\}$ will be called <u>evaluations</u>.

A Strategy Q will be called an <u>Evaluating Strategy</u> if and only if

$$s \in T_i \text{ implies } Q(s)(s) \in T_{i-1}.$$

The following theorem is a special case of Theorem 2.1, relevant to W-problems and hence to problem-like M-situations and establishes that $\bigcup_{i>0} T_i = T'$.

<u>Theorem 2.6</u> An evaluating strategy is a winning strategy for every member of $\bigcup_{i\geq 0} T_i$.

Proof:

Let $s \in \bigcup_{i>0} T_i$; then for some n, $s \in T_n$, $n > 0$.

If $n = 1$, then $Q(s)(s) \in T$ showing that Q is a winning strategy for T_1.

Let Q be a winning strategy for all $s \in T_i$. Let $s \in T_{i+1}$, then by definition of Q

$$Q(s)(s) \in T_i$$

Since Q is a winning strategy for T_i, there exists a sequence $f_1, f_2, \ldots f_i$ such that $f_i(f_{i-1} \ldots f_1(Q(s)\ (s)) \ldots) \in T$; and

$$f_k = Q(f_{k-1}(\ldots f(Q(s)(s)) \ldots) \text{ for each } k \leq i.$$

Hence Q is a winning strategy for s.

The next theorem shows that one can make the following statement, "If there is more than one evaluating strategy, then it is unnecessary to use any one of them consistently to arrive at a solution". Formally, it can be stated as follows.

Theorem 2.7 Let $\{S_\alpha\}$ be a partition of $\bigcup_{f \in F_0} S_f - T$. Let $\{Q_\beta\}$ be a set of evaluating strategies. Let K: $\{S_\alpha\} \to \{Q_\beta\}$ associate a strategy with each class of the partition. Denote by $K(S_\alpha)/S_\alpha$ the restriction of $K(S_\alpha)$ to S_α. Then $\bigcup_\alpha K(S_\alpha)/S_\alpha$ is an evaluating strategy.

Proof:

Let $s \in S_\alpha$. Then $\bigcup_\alpha K(S_\alpha)/S_\alpha(s) = K(S_\alpha)(s)$. Since $K(S_\alpha)$ is an evaluating strategy if $s \in T_i$, then $K(S_\alpha)(s)(s) \in T_{i-1}$. Hence for all $s \in T_i$,

$$(\bigcup_{\alpha} K(S_{\alpha})/S_{\alpha})(s)(e) \in T_{i-1},$$

proving $\bigcup_{\alpha} K(S_{\alpha})/S_{\alpha}$ is an evaluating strategy. That domain of $\bigcup_{\alpha} K(S_{\alpha})/S_{\alpha}$ is as required for a strategy follows for the definite of (S_{α}).

In view of this theorem, it is now not necessary to know the description of the equivalence classes of the Kernel of a specific evaluating strategy to apply it. All one needs to know is that some evaluating strategy is being applied. To assure oneself of that, one needs to know the descriptions of the sets T_i instead. Knowing these descriptions, one can find a winning solution for s by finding out that $s \in T_i$ and obtaining control $f \in F_o$ such that $f(s) \in T_{i-1}$. If F_o is a tractably small set, this can be done by enumeration. The resulting sequence of applications of controls will be according to some winning strategy and yield a solution. If F_o be infinite, there is no claim that this method of constructing winning situations is in any way realistic.

8. Strategies Based on T'

The construction of evaluating strategies depend strongly on having available the description of each set T_i in an evaluation. In some special cases, it may be possible to develop a strategy with a much smaller repertoire of descriptions. In what follows a method for strategy-construction will be discussed which merely needs a description of T'.

Given a W-problem $\langle S, F_0, T\rangle$ one can define a relation K' on S as follows

aK'b if and only if for some $f \in F_0$, $b = f(a)$. Let K be the transitive closure of K'. K is the property that if aKb, then one can change situation a to situation b by the successive applications of controls.

A W-problem will be called progressively finite if and only if

F1. K is irreflexive, i.e. no situation s is such that sKs (no "looping" is possible).

F2. There is no infinite chain $s_1, s_2, s_3 \ldots$ such that for each i $s_i K s_{i+1}$

F1 effectively says that each action taken on the way to solving a problem is "irrevocable". In a way this is a very comforting situation, since no matter how "blindly" one applies control, one never gets "caught ... loop".

F2 essentially says that the process of applying controls always reaches a "dead end". This prevents one "going on forever" on an "open-ended loop".

Neither the Tower of Hanoi problems nor propositional calculus described in Sec. 4 are progressively finite. There will be occasion to exemplify the analogs of progressively finite problems in the next Chapter

when the game of Nim is discussed. For the present only some formal properties of progressively finite problems will be discussed.

K, being a irreflexive and transitive is necessarily anti-symmetric. That is, it h the two essential properties of a partial order. Every chain in this order, being finite, has a lower bound. Hence the set of situations have a set of minimal elements.

The most useful thing about progressively finite W-problems is that a description of T' and (as a part of the problem specification) T is all that is needed to construct a winning strategy. To see this, one can define the following kind of a strategy.

A strategy Q: $\bigcup_{f \in F_o} S_f - T \to F_o$ is called <u>cautious</u> if and only if

$$s \in T' \text{ implies } Q(s)(s) \in T' \cup T$$

Evidently, since every evaluating strategy is a cautious strategy, cautious strategies exist. However, the important thing to note is that every cautious strategy is a winning strategy, whether it is an evaluating strategy or not, as long as the problem is progressively finite. This can be seen in terms of the two following theorems.

<u>Theorem 2.8</u> Let $\mathcal{P}$ be a progressively finite W-problem. Let S_o be the set of minimal elements of K. Then $S_o - T \subseteq S - T'$.

<u>Proof</u>:

$s \in S_o$ if and only if there is no $f \in F_o$ and no $s' \in S$ such that $f(s) = s'$. Hence

$$S_o = S - \bigcup_{f \in F_o} S_f$$

This leads to

$$S - T' \supseteq S - \bigcup_{f \in F_0} S_f \supseteq S - \bigcup_{f \in F_0} S_f = T = S_0 - T$$

Theorem 2.9 In a progressively finite W-problem every cautious strategy is a winning strategy for every element of T'.

Proof:

Let $s \in T'$ and Q a cautious strategy. Define a sequence $\{s_i\}$ of situations as follows:

$$s_0 = s$$

$$s_{i+1} = Q(s_i)(s_i) \text{ for all } i.$$

It is clear from the definition that for all i, $s_i K s_{i+1}$. Hence the sequence $\{s_i\}$ is finite. Let s_t be the minimal element of this chain.

If $s_i \in T$ for some $i \leq t$, then the sequence $Q(s_0), Q(s_1) \ldots Q(s_{i-1})$ is a winning sequence showing that Q is a winning strategy for s_0. If $s_i \notin T$ for all $i \leq t$, then, since Q is a cautious strategy $s_i \in T'$ for all $i \leq t$. Hence $s_t \in T'$ in particular. However $s_t \in S_0$, being a minimal element of K. Also $s_t \notin T$ whence $s_t \in S_0 - T \subseteq S - T'$ by theorem 2.7 which contradicts $s_t \in T'$.

It can be stated in a way analogous to the discussion at the end of the last section that for applying a cautious strategy, the description of the equivalence classes of its Kernel need not be known. If one has a situation $s \in T'$ and chooses $f \in F_0$ such that $f(s) \in T'$, one knows that some cautious strategy is being applied. To prove this rigorously one would have to prove an analog of theorem 2.7. This appears straight-forward and need not be belabored here. Indeed, some may even argue that all this

"rigorous rigmarole" (even if it is considered rigorous -- there are many small points slurred over in the discussion) does not yield any results that one could not be gleaned intuitively. Indeed, most rigorous discussions often take place only after some intuitive basis for them have been suggested. However, rigor has the advantage that through it one can clearly see the conditions under which the intuitively obtained results are valid. This gives a clearer insight into how an intuitively feasible operation may be improved when it is found to be unusable in reality. In our discussion of the General Problem Solver in the next two sections we shall endeavour to explain the many reasons for the occasional failures of the GPS.

9. Strategies Based on Subgoals -- the General Problem Solver [14]

To discuss methods based on subgoals, it will be necessary to discuss evaluations based on sets other than T. One of the subgoal-types used in the GPS is "Apply operator f to situation s". This is a trivial operation if $s \in S_f$. Otherwise, one sets up the subgoal "Transform s so that f can be applied". This is equivalent to solving a new problem, with S and F_o the same as before but with T replaced by S_f. Any solution for this new problem may be discussed in terms of evaluations. It is probably not essential to use the idea of evaluations. However, at the present level of the author's understanding, any concept more general than evaluations is apt to be hard to handle. Moreover, workers using the idea of subgoals often have the idea of "reducing differences" implicity in their argument. So the use of evaluation as a cornerstone of the theory of subgoals will probably not be an inherent limitation on the way workers in the field interpret the term "subgoals".

Given a W-problem and a subset $X \subseteq S$ one defines a class of sets X_i as follows

$$X_o = X$$

and for all $i > 0$

$$X_{i+1} = \left\{ s \mid s \notin \bigcup_{k=0}^{i} X_k \text{ and } (\exists f)(f \in F_o \text{ and } f(s) \in X_i) \right\}$$

As in section 7, we shall denote $\bigcup_{i>0} X_i$ by X'.

One now defines a set of subsets $(S_{fX})_{f \in F_o}$ of X', indexed by F_o, as follows

$$s \in S_{fX} \text{ iff for some } i > 0$$

$s \in X_i$ and $f(s) \in X_{i-1}$. The $(S_{fX})_{f \in F_o}$ are not necessarily pairwise disjoint. However $\bigcup_{f \in F_o} S_{fX} = X'$. The class of sets $(S_{fX})_{f \in F_o}$ is a cover, rather than a partition on X'.

The GPS sets up the goal "Apply operator f to situation s" in view of the recognition of certain differences between the winning set (either T or S_g for same $g \in F_o$). The extraction of this difference does not assure that $s \in S_f$. For the purposes of the present discussion the set S^o_{fX} will denote the set of all situations s such that if the sub-problem is "transform s to X", the the subgoal, "apply f to x" will be set up.

Although the GPS is a scheme for directed search for solution, one can envisage "difference tables" in GPS which give rise to minimum search. A GPS-like algorithm will be quoted late in this section which would be effective on such an optimal decision table. Our main purpose in this section will be to set up certain conditions on the structure of the difference table (the sets S^o_{fX}) which are sufficient for the successful convergence of that algorithm. The sufficiency will be exhibited with a series of lemmata. Later on there will be occasion to discuss how one can make modifications of the given algorithms to an exact replica of the GPS. It will be indicated how the convergence can be assured even for a slight relaxation of the axioms on S^o_{fX}. The axioms regarding the sets S^o_{fX} will make explicit certain assumptions which are either tacitly made or hoped for in literature about the difference tables. It is considered useful to bring these "out in the open".

One assumes initially, of course, that the difference table is such that if a situation can be transformed into a winning situation, the difference table will indicate some transformation for it. This is reflected in axioms D1 and D3 below. Also, if a certain transformation is

indicated, then if the transformation is applicable, the "distance" between the situation and the winning states is actually reduced. This is indicated in D2 below. This reduction may be considered inessential and it may be possible to prove convergence for a more relaxed condition: for the present this assumption is made as a member of sufficient conditions only.
Another important assumption (which, perhaps, may also be relaxed) indicates that if the application of a certain transformation f is indicated, then any other transformation used for making f applicable does not carry the situation away from the winning set.

One reason for setting up the assumptions formally is to indicate that the convergence of the GPS is difficult to assure intuitively. Hence if relaxed assumptions are envisaged on intuitive grounds, the proof of the convergence of GPS will have to be carried out with a certain standard of rigor.

One concentrates on the following class of sets

$$D = \{T\} \cup \{S_f \mid f \in F_o\} .$$

A class of sets $\{S^o_{fX} \mid f \in F_o, X \in D\}$ is now defined with the following properties

D1. For each set $X \in D$ and each $f \in F_o$, $X' = \bigcup_{f \in F_o} S^o_{fX}$.

D2. $S^o_{fX} \neq \emptyset$ implies

$$\emptyset \neq S^o_{fX} \cap S_f \subseteq S_{fX} \text{ for each } f \in F_o,\ X \in D.$$

D3. $S^o_{fX} - S_f \subseteq \bigcup_{g \in F_o} S^o_{gS_f}$

D4. $s \in (S^o_{fX} - S_f) \cap X_i$, and $s \in S_g$ implies $g(s) \in S^o_{fX} \cap X_j$ where $j \leq i$.

An algorithm modelling the GPS (together with suggestions for making the model more realistic) will be given presently. Meanwhile, the following sequence of lemmata will indicate the convergence of the algorithm as presented.

Lemma 2.10

$$S^o_{fS_f} = \emptyset$$

Proof

$S^o_{fS_f} \cap S_f \subseteq S_{fS_f}$ by D2. Hence, intersecting both sides with S_f

$$S^o_{fS_f} \cap S_f \subseteq S_{fS_f} \cap S_f.$$

However, by definition $S_{fS_f} \subseteq S_f^!$ and $S_f^! \cap S_f = \emptyset$. Hence $S^o_{fS_f} \cap S_f = \emptyset$. Contrapositive of D2 yields $S^o_{fS_f} = \emptyset$.

Given an element $s \in T'$ one can set up a sequence $\{X^i(s)\}$ of elements of D (called a difference sequence) as follows:

$$X^o(s) = T.$$

Since $s \in T'$ there is an element $f \in F_o$ such that $s \in S^o_{fX^o(s)}$ by D1. $X^1(s)$ is defined to be S_f. For all $i \geq 1$, $X^{i+1}(s)$ is defined if and only if $s \notin X^i(s)$. In this case $s \in S^o_{fX^{i-1}(s)} - X^i(s)$ where $X^i(s) = S_f$ for some $f \in F_o$. By D3 there exists a $g \in F_o$, such that $s \in S^o_{gX^i(s)}$. $X^{i+1}(s)$ is then defined to be S_g. Clearly $g \neq f$, since in this case $s \in S^o_{fS_f}$ which contradicts lemma 2.10 above.

A difference sequence is said to end at i if $X^{i+1}(s)$ is undefined.

Lemma 2.11

If the sequence $\{X^i(s) | i = 0,1,\ldots n\}$ ends at n, then $X^n(s) \neq X^i(s)$ for any $i < n$.

Proof

If $X^{n+1}(s)$ is undefined, then $s \in X^n(s)$. If $X^n(s) = X^i(s)$, then $s \in X^i(s)$ whence $X^{i+1}(s)$ is undefined, contradicting the hypothesis.

The difference chain $\{$sequence$\}$ for an element $s \notin T'$ is not necessarily unique. However, the extra assumption will be made that

D5. There is an integer N such that for each $s \in T'$, all difference sequences end at some $i \leq N$.

Lemma 2.12

Let $\{X^i(s) | i = 1,2,\ldots n\}$ be a difference sequence for s, ending at n. Let $X^n(s) = S_f$, and $X^i(s) = S_{g_i}$ for all $i > 1$. Then $f(s) \in S^o_{g_i X^{i-1}(s)}$ for each $i > 1$.

Proof

By lemma 2.11 $f \neq g_i$ for all $i(1 < i < n)$. Also, by definition of difference sequence $s \in S^o_{g_i X^{i-1}(s)} - S_{g_i}$ for each $i(i \leq i < n)$. Again by D1, $s \in (X^{i-1}(s))'$ whence $s \in X^{i-1}(s))_j$ for some j. Hence $s \in (S^o_{g_i X^{i-1}(s)} - S_{g_i}) \cap (X^{i-1}(s))_j$. Since $f \neq g_i$, by D4 $f(s) \in S^o_{g_i X^{i-1}(s)}$.

Lemma 2.13

Under the hypothesis of Lemma 2.12, if $s \in (X^i(s))_{j_i}$ for each i, then $f(s) \in (X^i(s))_{k_i}$ where $k_i \leq j_i$.

The proof follows a-forteriori from the proof of Lemma 2.12

Let $\{X^i(s)\}$ be a difference sequence for s ending at n. Then $s \in S^o_{g_i X^{i-1}(s)}$ for all $i \leq n$. We can prove the following rather crucial lemma.

Lemma 2.13

For $s \in S^o_{g_i X^{i-1}(s)}$, either $s \in X^i(s) = S_{g_i}$ or there is finite

sequence $h_1, h_2, \ldots h_k (h_i \in F_o)$ such that $h_k(h_{k-1}(\ldots h_1(s)\ldots)) \in S^o_{g_i X^{i-1}(s)}$ $\cap S_{g_i}$.

<u>Proof</u>

This is trivially true if $i = N$, since $s \in S^o_{g_N X^{N-1}(s)} \cap S_{g_N}$ by D5. For $i = N-1$, if $s \in S_{g_{N-1}}$, then $s \in S_{g_N X^{N-1}(s)} \cap S_{g_N} \subsetneq S_{g_N X^{N-1}(s)}$. Also from the proof of lemma 2.12, $s \in (X^{N-1}(s))_k$ for some k. Hence $g_N(s) \in (X^{N-1}(s))_{k-1}$ by definition of $S_{g_N X^{N-1}(s)}$. Also, be lemma 2.12 $g_N(s) \in S^o_{g_{N-1} X^{N-2}(s)}$. Hence there exists a g'_N (by D3 and D5) such that $g_N(s) \in S^o_{g'_N X^{N-1}(s)} \cap S^o_{g'_N}$ whence $g'_N(g_N(s)) \in (X^{N-1}(s))_{k-2}$. By k repetitions of this process one obtains an element as indicated in the theorem.

Let now this theorem be true for $i = k$. Let $s \in S^o_{g_{k-1} X^{k-2}(s)}$. If $s \notin X^{k-1}(s) = S_{g_{k-1}}$ then by D3, $s \in S^o_{g_k X^{k-1}(s)}$. Either $s \in S_{g_k}$ or there is a finite sequence $\{h_i | h_i \in F_o \; i \leq n\}$ such that $h_n(h_{n-1} \ldots h_1(s)\ldots) \in S_{g_k}$. If for some j, $h_j(h_{j-1} \ldots h_1(s)\ldots) \in X^{k-1}(s)$ then the theorem is proved. Otherwise, let $s \in (X^{k-1}(s))_m$. Then $g_k(h_n(h_{n-1} \ldots h_1(s)\ldots) \in (X^{k-1}(s))_{m-1}$. By a finite repetition of this process one eventually arrives at an element $s' \in X^{k-1}(s)$.

One can now state the basic result of this section. Let $s_o \in T'$. By considering the above lemmata one can see that if all the elements of a class of sets $\{S^o_{fX} | f \in F_o,\ X \in D\}$ can be recognized, then the following process will generate a winning solution for s_o in a finite number of steps.

1. Set $k = 1$, $j = 0$, $i = 0$, let $X^i = T$.
2. If $s_j \in X^i$, go to step 4.
3. If $s_j \notin X_i$, find f such that $s_j \in S^o_{fX^i}$ set $i = i+1$, $X^i = S_f$. Return to step 2.
4. If $X^i = T$, Stop.
5. If $X^i = S_f$, set $j = j+1$, $s_j = f(s_{j-1})$, set $(\text{funct})_k = f$; set $k = k+1$, $i = i-1$. Return to Step 2.

A class $\{S^o_{fX} | f \in F_o, X \in D\}$ always exists. If $S^o_{fT} = S_{fT}$ and $S^o_{fS_g} = S_{fS_g}$ for all f, $g \in F_o$, one has $T' = \bigcup_{f \in F_o} S_{fT}$ satisfying D1. D2 is satisfied since $S_{fX} \subseteq S_f$ for all X. D3 is satisfied since $S_{fX} - S_f$ is empty. D4 is also satisfied since the antecedent is false. D5 is satisfied for $N = 1$ since $X'(s) = S_f$ implies $s \in S_{fT} \subseteq S_f$ hence $X^2(s)$ is undefined. However, the set of classes of sets $\{S^o_{fX} | X \in D, f \in F_o\}$ may be much richer than the consisting of only the class $\{S_{fT} | f \in F_o\} \cup \{S_{fS_g} | f, g \in F_o\}$. Hence for some of these classes $\{S^o_{fX}\}$ may be easier to describe in a given language than $\{S_{fT}\}$. Hence, in spite of the fact that the $\{S_{fT}\}$ are (at least conceptually) constructively defined and the $\{S^o_{fX}\}$ are not (D1-D5 are far from constructive definitions) so definable at present, does not perclude their usefulness.

The above discussion is intended to form a model for finding solutions which have very close analogies with the General Problem Solver. The intention was to set up the S^o_{fX} as the Kernels of maps mapping every point to some specific difference with X. It will be noticed that $S^o_{fX} \cap S^o_{gX} (g \neq f)$ is not necessarily empty. It is this fact which gives rise to the non-uniqueness of difference sequences. If one relaxes the condition

D5 to read, "for each $s \in T'$, at least one difference sequence ends at some $i \leq N$" and if the number of non-disjoint S^o_{fX} for a given X is finite a small modification can be made in the procedure to find such a difference sequence. It appears from a perusal of the flow-charts of the GPS that the occasional "back-ups" are caused by such a search. The search can be cut back even further by the fact that it is useless to have $X^j(s) = X^i(s)$ in any difference sequence, by virtue of the following lemma.

Lemma 2.14

If $\{X^i(s)\}$ is a difference sequence where $X^j(s) = X^k(s) (j > k)$, then $\{Y^i(s)\}$ is also a difference sequence, where

$$Y^i(s) = X^i(s) \text{ for } i \leq k$$
$$Y^i(s) = X^{(i-j)+k}(s) \text{ for } k > j$$

The proof will be left to the reader. The point that is to be made is the above model need not be the most faithful model of the GPS and that more faithful models can be built. However, no matter what the model is, it may be worthwhile to consider the exact conditions (like D1 to D5 above) under which the model can be used for finding winning solutions.

When a GPS-like program meets with occasional failure, the need arises to modify the distance-transformation table. Such modifications may be made in a directed manner if one can pinpoint the failure to one or other of the conditions required for the convergence of the procedure for constructing winning solutions. Of course, to enable such a process, it is necessary to understand the basic structure of the W-problem involved. Often this structure is ill-understood and difficult to understand. No attempt is being made here to ignore the difficulty. However, the basic

structures presented here do indicate some well-directed avenues which help to decide the directions in which understanding should be attempted.

10. Sundry Remarks Regarding Search for Winning Sequences

In Sections 7 and 8 methods were developed for defining strategies in such a way that the use of the strategy did not need the recognition of the elements of their Kernel. In Section 9, however, what was described was a procedure for constructing a winning sequence. It will be noticed that for finding an element f_k of the winning sequence one not only used the point s, but also kept in mind the previous procedures used in finding f_k (notice that in Step 5, i was set to i-1 after the finding of f_k, not to 1). It is possible that one can use GPS-like procedures for defining strategies: (for instance if $f \neq g$ implied $S^o_{fX} \cap S^o_{gX} = \emptyset$ for all X, one could easily transfer control to 1 in the procedure) however, it is not clear that one should limit oneself to the concept of strategies as a method for solution construction.

Two remarks connected with the Tower of Hanoi come to mind here. The reader may verify that the following procedure generates a winning solution for $< (1,2,3,\ldots n), \emptyset, \emptyset >$.

1. Apply $(n,(-1)^{n+1})$
2. If the resulting state $= < \emptyset,(1,2,\ldots n),\emptyset >$, Stop.
3. In the resulting state $< (X_{01}\ldots X_{0i_0})(X_{11},\ldots X_{1i_1})$
   $(X_{21},\ldots X_{2i_2}) >$.

Let $X_{ki_k} = n$. Find $\max_{m,j \neq k} (X_{ji_j}, X_{mi_m}) = X_{ti_t}$. Apply (X_{ti_t}, s), where s is such that $t+s \pmod 3 \neq k$. Return to step 1.

In words, "Move the smallest disc. Then make the only move possible without moving the smallest disc. Go back to moving the smallest disc. Always move the smallest disc in the same direction".

This procedure does not yield the winning solution for all situations, once in a while getting into a "loop". However, for the cases where it does yield a solution, the solution coincides with the one obtained by the previous strategy of Section 6.

In neither of the above cases was it clear as to how the procedure for obtaining the winning solution was discovered. The following considerations seem to lead to a more "natural" way of obtaining a solution.

One realizes that, if the situation $<(1,2,\ldots n),\emptyset,\emptyset>$ has to be coverted into the situation $<0, (1,2,\ldots n),\emptyset>$ by applying controls as restricted by the rules given, it is necessary that 1 be "moved" from the first to the second position and for this all the other discs have to be in the third position. Hence one must, sometime in the course of applying the controls, obtain the state $<(1),\emptyset,<2,3,\ldots n>>$, which then is changed to $<\emptyset,(1),(2,3,\ldots n)>$. It is also clear, that now the set of discs $(2,3,\ldots n)$ can be moved back to the second position: one can apply a sequence of controls very similar to the ones needed to move this set from the first to the third position. It is also clear that the problem of going from $<(1,2,\ldots n),\emptyset,\emptyset>$ to $<(1),\emptyset,(2,3,\ldots,n)>$ is analogous to the problem of going from $<(1,\ldots n-1),\emptyset,\emptyset>$ to $<\emptyset,\emptyset,(1,2,\ldots n-1)>$. These conditions lead naturally to the setting up a recursive procedure, to be called, "Move $(p,p+1,\ldots n)$ from position k to position ℓ" ($k=0,1,2$; $\ell=0,1,2$; $\ell\neq k$)". The procedure is as follows:

"If $p=n$, move n from position k to position ℓ (apply (n,t) where $k+t = \ell$(mode 3)). If $n\neq p$, move $(p+1,\ldots n)$ p from position k to position m ($m\neq k,\ell$). Move p from position k to position ℓ. Move $(p+1,\ldots n)$ from position m to position ℓ".

The overall procedure then is "Move $(1,2,\ldots n)$ from position 0 to position 1".

What has been done above is to set up the idea of a "Macro-control" $M(p,p+1,\ldots,n;\ k,\ \ell)$ which consists of sequences of elementary controls. The sequence, as shown above in words, is generated by the following definitions.

1. $M(p;k;\ell) = (p,t)$ where $k+t = \ell \pmod 3$
2. $M(p,p+1,\ldots n;\ k;\ \ell) = M(p+1,\ldots,n;\ k;\ m)(p,t)M(p+1,\ldots,n;\ m;\ \ell)$ where $m \neq k,\ell$

If one now expands $M(1,2,3;0;1)$ to generate a winning sequence for $<(1,2,3),\emptyset,\emptyset>$ one obtains the following sequence

$$\begin{aligned} & M(1,2,3;0;1) \\ &= M(2,3;0;2)(1,+1)M(2,3;2;1) \\ &= M(3;0;1)(2,-1)M(3;1;2)(1,+1)M(3;2;0)(2,-1)M(3;0;1) \\ &= (3,+1)(2,-1)(3,+1)(1,+1)(3,+1)(2,-1)(3,+1) \end{aligned}$$

yielding the same solution as in Section 4.

From here, it is a matter of perseverance to show why the strategy of Section 6 is a natural consequence of the above recursive procedure. This will not be attempted here. However, the point has to be made that since the function $M(1,2,\ldots n;k;\ell)$ was not in the original repertoire of elementary statements used in describing the rules of the game, this does not throw any light on how one can mechanically generate this function from the original rules. For a discussion of these points see Amarel [15].

However, all these considerations do shed some light on what is often called "Method of Subgoal generation" in literature. Here the term will be used somewhat more strictly then is often used, to give some concrete meaning to the discussion. The discussion, however, will remain informal.

Given a W-problem $< S,F_oT >$ and an element $s_o \in T_j$, $k < j$, T_k is called a <u>subgoal</u> for s_o. If $k < m < j$ and $T_m = \{s'\}$, a unit set, then the pair (s',T_k) is called a <u>sub-problem</u> for s_o. In the above case of the tower of Hanoi, the initial situation is an element of $T_{2^{n-1}}$; the only element of the $T_{2^{n-1}-2}$ that need be considered is $< (1),\emptyset,(2,3,\ldots n) >$. The attainment of this subgoal $T_{2^{n-1}-2}$ leads to two successive sub-problems $(< (1),\emptyset,(2,3,\ldots n) >, T_{2^{n-1}-1})$ where $T_{2^{n-1}-1} = \{<\emptyset,(1),(2,3,\ldots n) >\}$ and $(< \emptyset,(1),(2,3\ldots n) >, \{< \emptyset,(1,\ldots n)\emptyset >\})$. The advantage of this kind of breaking up of a problem into sub-problems is evident in the case of the Tower of Hanoi -- the successive steps in the breaking up exhibits the entire winning solution! As to whether this is feasible in all cases depends heavily on whether the language of discussion is strong enough to indicate that some of the sub-goals are unit sets, i.e., when there are unit subgoals. Very little research has been done in this area, even though words like subgoal and 'sub-problem' have been around ever since the inception of the field. The reason may very well have been that the advantages of precision of definitions have been consistently overlooked.

It ought to be pointed out that the idea of subgoals has meaning even when subgoals are not unit sets. The reason for this is as follows. If $s_o \in T_j$ and $m < j$ and F_o is a finite set of cardinality, say, k. Then there are at most k^{m-j} possible control sequences, at least one of which leads to a situation in T_m. From this situation, at least one of at most k^m sequences leads one to a winning situation. Hence if s_o can be recognized to be in T_j and T_m can be recognized, a total of at most $k^m + k^{j-m}$ systematic searches are necessary instead of the k^j searches that would otherwise required in the absence of any knowledge about the T_j. Of course, all this search would be unnecessary if T_i could be recognized for each i.

In the previous sections, the entire idea of searches have been avoided. As indicated above, searches become necessary when any of the techniques discussed in the previous sections (or similar techniques not discovered yet) cannot be applied due to ones inability to recognize some of the sets involved.

At present very little is understood about optimal search procedures in the context discussed in this book. In what follows, a very informal approach will be made towards setting up some ideas on the basis of which search may be discussed. However, it ought to be pointed out at the outset that search cannot be carried out with confidence -- even in principle -- if there is no method for recognizing T'.

Given a W-problem (S,F_o,T) one can associate with each $s \in S$ a set, $F(s) = \{f \mid f \in F_o \text{ and } s \in S_f\}$; $F(s)$, of course is given by the rules of the problem. $K(s)$ will denote the set of all linear well-orderings on $F(s)$. Clearly, any element of $K(s)$ is a subset of $F(s) \times F(s)$ and hence a subset of $F_o \times F_o$. Let $B(F_o \times F_o)$ denote the set of all subsets of $F_o \times F_o$, i.e., the set of all binary relations on F_o. By a <u>search strategy</u> will be meant a function

$$S_T : S \to B(F_o \times F_o)$$

such that $S_T(s) \in K(s)$ for every s. For some elements of s (when $s \notin \bigcup_{f \in F_o} S_f$) $S_T(s)$ is the empty ordering.

On the assumption that F_o is finite and T' is finite and recognizable and given any search strategy S_T one can set up the following procedure for constructing a winning strategy for $s_o \in T'$.

1. Set $i=0$
2. Set $X_i = S_T(s_i)$
3. If $s_i \in T$ stop: indicating success

4. If $s_i \notin T'$ set $i=i-1$ if $i > 0$ and return to step 6. Otherwise stop indicating failure.

5. If $s_i = s_j$ for some $j < i$ set $i=i-1$ if $i > 0$ and return to step 6. Otherwise stop indicating failure.

6. If X_i is empty set $i=i-1$ if $i > 0$ and return to step 6. Otherwise stop indicating failure.

7. Set $(\text{funct})_i$ = least element of X_i. Subtract $(\text{funct})_i$ from X_i and store result as X_i. Set $i=i+1$, $(\text{funct})_{i-1}(s_{i-1}) = s_i$. Return to step 2.

Such a procedure (an exhaustive search determined by S_T) would stop after a finite time indicating success for all $s_o \in T'$. In a progressively finite problem (see Section 8) T' need not be finite for completion of the procedure. However, the crucial point here is the fact that this finite procedure may turn out to be impossibly long if S_T is not well chosen. If $S_T(s)$ turns out to be such that its least element turns out to be f where $s \in S_{fT}$, then an extremely rapid process will result.

One can say somewhat imprecisely that most methods developed by workers in the field consists of setting up efficient search strategies. Most methods dealing with problems of the W-problem type tend to set up sets like S_{fT}, S^o_{fX}, T_i or T' as described in previous sections. These are set up generally from common sense (or "learned" -- as later chapters will indicate). However, it is kept in mind that the sets "guessed at" may not coincide with what they are supposed to be so that when a certain control applied to a certain situation does not lead to a winning situation one can "start over again" using a different control. This leads to something in the nature of a search-strategy. In many cases, it has turned out that

the search strategy so induced is better than what can be expected from an arbitrarily chosen search-strategy.

Search strategies in literature are often based on what are called "Intermediate Evaluations". In the point of view adopted in this book, the Intermediate Evaluation Functions are merely alternative ways to form descriptions of the $\{T_i\}$ or of T'. This will be discussed at some length in Chapter III. It will suffice to point out here that since Intermediate Evaluations are functions with the set of situations as their domain, their Kernels defines partitions on the situation-set $\mathcal{S}$.

There is a general belief often expressed in the literature that problems can be best attacked by "going backwards" from the winning situation, i.e., by successively generating members of T_1, T_2, T_3, etc., till s_o is located in some T_i. This belief would be valid if these sets did not grow exponentially with i. For instance, if the problem had the structure of a tree rooted at s_o (i.e., if all situations $s \neq s_o$ was such that $s = f(s')$ for an unique f and s') then such a generation method would be highly efficient. Exactly the opposite case would occur if each situation in T' was a member of S_f for only one f and each situation s was such that for each $f \in F_o$, there was a situation s' such that $f(s') = s$. Here, it might be better to "go forward".

The above paragraph indicates that search processes based on enumeration of situations can only succeed in very special cases. Methods for recognizing such special cases have not been developed. Nor have many methods been developed for constructing descriptions of the sets discussed in Sections 7, 8 and 9 from the description of the problem. It is becoming increasingly clear that the use of the proper description language is a very crucial matter here. This will be discussed in somewhat greater depth in later chapters.

Before concluding this chapter and section it may be worthwhile to point out that although a study of efficiencies of search strategies has not been made in a rigorous way, it may be extremely worthwhile doing. As clearer understandings develop of the sets discussed in this chapter and more are added to this repertoire, the effect of errors in recognizing these sets may be come clearer.

CHAPTER III - GAMES AND SOLUTION METHODS

1. Introduction

In the previous chapter M-situations were introduced as formal structures and the ideas of forcing situations and neutral situations were introduced, as well as the idea of winning and non-losing strategies. Also, an important theorem (Theorem 2.1) was quoted regarding the existence of winning and non-losing strategies.

The general model above was then specialized to yield a class of structures which had one-one correspondences to W- problems. Also, it was shown that winning strategies of these special class of M-situations could be utilized for constructing W-strategies for the corresponding W- problems and hence for constructing winning solutions. It was indicated how W- problems are adequate formal models for many problems studied in the field of Artificial Intelligence. A number of alternative methods of constructing winning solutions for W-problems were then discussed.

A similar sequence of discussions will be undertaken in the present chapter, dealing with the formal model of a wide class of two-person board games. As is well-known the classical model of games can be specialized to cover exactly the same situations. Many of the formal notions introduced will be superficially analogous to those introduced in Chapter II: however, it is not clear that the ideas in Chapter II would be special cases of ideas developed in the present chapter. Such relationships will not be discussed. Also, as before, no attempts will be made to derive results as special cases of results obtainable for M-situations -- even though it may be possible in some cases.

In the next section the special class of M situations will be introduced as models of game situations and the special properties of winning strategies pertaining to these models will be discussed. In Sec. 3 specific board games will be formalized to conform to the structure of these special classes. In later sections methods for construction of winning strategies will be considered. Here, as in the previous chapters, the importance of the language for describing certain sets of situations will be kept in mind.

2. Game Situations and Strategies

A Basic Game Situation is an M-situation $\langle S, C, D, M, S_W, S_L \rangle$ with two pre-specified elements $c_o \in C$ and $d_o \in D$ such that

G1) $S_c \cap S_d \neq \emptyset$ implies either $c = c_o$ or $d = d_o$ but not both.

The following facts are worth noticing

Lemma 3.1

$$S_{c_o} = \bigcup_{d \neq d_o} S_d; \quad \text{ii)} \quad S_{d_o} = \bigcup_{c \neq c_o} S_c$$

Proof: Let $s \in S_{c_o}$, then by definition there exists a $d \in D$ and $s' \in S$ such that $(s, c_o, d, s') \in M$ whence $s \in \bigcup_{d \in D} S_d$. However $s \notin S_{d_o}$ since $S_{c_o} \cap S_{d_o} = \emptyset$ by G1. Hence $S_{c_o} \subsetneq \bigcup_{d \neq d_o} S_d$.
Conversely, if $s \in S_d$ where $d \neq d_o$, then by the same argument as above there exists a c such that $s \in S_d \cap S_c$. However since $d \neq d_o$, by G1 $c = c_o$. Hence $s \in S_{c_o}$. Hence $S_{c_o} \supsetneq \bigcup_{d \neq d_o} S_d$.

The second part follows similarly

Lemma 3.2

i) $s \in S_{c_o}$ and $c \neq c_o$ implies $s \notin S_c$

ii) $s \in S_{d_o}$ and $d \neq d_o$ implies $s \notin S_d$

Proof: By lemma 3.1 $s \in S_c$ and $c \neq c_o$ implies $s \in S_{d_o}$, but this implies $s \in S_{c_o} \cap S_{d_o}$ which contradicts G1.

The second part follows similarly

The above lemmata indicate that $\bigcup_{c \in C} S_c$ has a partition consisting of S_{c_o} and $\bigcup_{c \neq c_o} S_c$. The same set $\bigcup_{c \in C} S_c$ coincides by definition of M-situations with $\bigcup_{d \in D} S_d$. This in its turn has a partition coinciding with the previous partition, S_{d_o} coinciding with $\bigcup_{c \in C} S_c$ and $\bigcup_{d \neq d_o} S_d$ coinciding with S_{c_o}.

The idea of basic game situation will further be specialized for our purposes in the following way: A basic game situation will be called a <u>game situation</u> if it obeys the following additional axioms.

G2) $S_W \cup S_L \subseteq S - \bigcup_{c \in C} S_c = S - \bigcup_{d \in D} S_d$

G3) $s \in S_{c_o} \cap S_d$ implies $(c_o, d)(s) \in S - S_W - S_{c_o}$

G4) $s \in S_c \cap S_{d_o}$ implies $(c, d_o)(s) \in S - S_L - S_{d_o}$

In effect the axioms say the following. "The players play alternately (c_o and d_o standing for "inaction"). The game stops whenever a win or a loss is reached (it can also stop in a draw -- see definition of S_D in Chapter II). The opponent ("disturbance") cannot make a final move into a win and the player cannot make a final move into a loss".

It is possible that the major points that will be made about game situations can be made with much weaker assumptions than made here. However, this fact will not be emphasized further in this book.

As in the previous chapter, the next paragraph will introduce a somewhat simpler-looking structure which will have many properties in common with game situations.

A <u>board game</u> is given by the 5-tuple $\langle S, G, F, W, L \rangle$ where S is an abstract set, F and G sets of functions from subsets of S into S and W, L subsets of S, with the following properties

B1) $\left(\bigcup_{f \in F} S_f\right) \cap \left(\bigcup_{g \in G} S_g\right) = \emptyset$

B2) $W \cap L = \emptyset$

B3) $W \cup L \subseteq S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$

B4) $s \in S_f$ and $f \in F$ implies $f(s) \in S - \bigcup_{f \in F} S_f - L$

B5) $s \in S_g$ and $g \in G$ implies $g(s) \in S - W - \bigcup_{g \in G} S_g$

Given a game situation $R = \langle S, C, D, M, S_W, S_L \rangle$ one defines a 5-tuple $B(R) = \langle S, F, G, W, L \rangle$ as follows

i) $W = S_W$; $L = S_L$

ii) For each $c \in C$, there is an unique element f_c F such that $f_c = (c, d_o)$ and these are the only members of F. For each d D there is an unique element g_d G such that $g_d = (c_o, d)$ and these are the only members of G. f_{c_o} and g_{d_o} are not defined since $(c_o, d_o)(s)$ is not defined.

Theorem 3.3 Given a game situation R, B(R) is a board game.

Proof: S is a set and W and L are subsets of S as required by the definition of a board game. B2 is satisfied since S_W and S_L are disjoint by definition of an M-situation. It is clear from the construction of B(R) that $\bigcup_{f \in F} S_f = \bigcup_{c \in C} S_c \cap S_{d_o}$. However, since $S_{d_o} = \bigcup_{c \neq c_o} S_c$ by lemma 3.1, one has $\bigcup_{f \in F} S_f = \bigcup_{c \neq c_o} S_c$. Similarly $\bigcup_{g \in G} S_g = \bigcup_{d \neq d_o} S_d$. Hence $\left(\bigcup_{f \in F} S_f\right) \cap \left(\bigcup_{g \in G} S_g\right) = \bigcup_{\substack{c \neq c_o \\ d \neq d_o}} (S_c \cap S_d) = \emptyset$ proving B1.

$$\text{Now } \bigcup_{c \in C} S_c = S_{c_o} \cup \left(\bigcup_{c \neq c_o} S_c\right) = \left(\bigcup_{d \neq d_o} S_d\right) \cup \left(\bigcup_{c \neq c_o} S_c\right)$$

$$= \left(\bigcup_{f \in F} S_f\right) \cup \left(\bigcup_{g \in G} S_g\right)$$

$$\text{Similarly } \bigcup_{d \in D} S_d = \left(\bigcup_{f \in F} S_f\right) \cup \left(\bigcup_{g \in G} S_g\right)$$

whence G2 reduces to B3.

Again, $s \in S_c$ implies $s \in S_c \cap S_{d_o}$ and $f_c(s) = (c, d_o)(s)$. By G3, $f_c(s) \in S - S_W - S_{d_o}$ $= S - W - \bigcup_{f \in F} S_f$ proving B4. Similarly G4 yields B5.

Given a board game $B = \langle S, F, G, W, L \rangle$ one defines a 6-tuple $R(B) = \langle S, C, D, M, S_W, S_L \rangle$ as follows:

i) $S_W = W$; $S_L = L$

ii) For each $f \in F$ there is an unique element $c_f \in C$. In addition there is an element $c_o \in C$. These are the only elements of C. Similarly the only elements of D are d_o and an unique element d_g for each $g \in G$.

iii) $(s, c, d, s') \in M$ if and only if

either a) $c = c_o$, $d = d_g$ for some $g \in G$ and $s' = g(s)$

or b) $c = c_f$ for some $f \in F$, $d = d_o$ and $s' = f(s)$

The following lemma is useful for tying together the next theorem.

Lemma 3.4 If B is a board game $\langle S, F, G, W, L \rangle$ then the following is true for R(B)

i) $S_{c_o} = \bigcup_{g \in G} S_g = \bigcup_{d \neq d_o} S_d$

ii) $S_{d_o} = \bigcup_{f \in F} S_f = \bigcup_{c \neq c_o} S_c$

iii) $\bigcup_{c \in C} S_c = \left(\bigcup_{f \in F} S_f\right) \cup \left(\bigcup_{g \in G} S_g\right) = \bigcup_{d \in D} S_d$

Proof of i

Let $s \in S_{c_o}$. Then there exists $d \in D$, $s' \in S$ such that $(s, c_o, d, s') \in M$. By construction $d = d_g$ for some $g \in G$ and $s' = g(s)$ so that $s \in S_g \subseteq \bigcup_{g \in G} S_g$. If $s \in S_g$ then $(s, c_o, d_g, g(s)) \in M$ whence $s \in \bigcup_{d \neq d_o} S_d$. Hence $S_{c_o} \subseteq \bigcup_{g \in G} S_g \subseteq \bigcup_{d \neq d_o} S_d$.

Again, let $s \in \bigcup_{d \neq d_o} S_d$; hence in particular $s \in S_{d_g}$. There exists $c \in C$ & $s' \in S$ such that $(s, c, d_g, s') \in M$. Hence $c = c_o$ i.e. $s' = g(s)$ whence $s \in \bigcup_{g \in G} S_g$. Also if $s \in S_g$, then $(s, c_o, d_g, g(s)) \in M$ whence $s \in S_{c_o}$. Hence $S_{c_o} \supseteq \bigcup_{g \in G} S_g \supseteq \bigcup_{d \neq d_o} S_d$.

ii) is proved similarly

iii) follows since $\bigcup_{c \in C} S_c = S_{c_o} \cup \bigcup_{c \neq c_o} S_c = \left(\bigcup_{g \in G} S_g\right) \cup \left(\bigcup_{f \in F} S_f\right) =$

$\bigcup_{d \neq d_o} S_d \cup S_{d_o} = \bigcup_{d \in D} S_d$.

Theorem 3.5 Given a board game B, R(B) is a game-situation.

Proof: Let $(s,c,d,s') \in M$ and let $(s,c,d,s'') \in M$. If $c = c_o$, and $d = d_g$, then $s' = g(s) = s''$. If $c = c_f$, $d = d_o$ then $s' = f(s) = s''$. This proves M1. To prove M2 assume $s \in S_c \cap S_d$. By lemma 3.4 $\left(\bigcup_{d \neq d_o} S_d\right) \cap \bigcup_{c \neq c_o} S_c = \left(\bigcup_{f \in F} S_f\right) \cap \left(\bigcup_{g \in G} S_g\right) = \emptyset$ by B1; either $c = c_o$ or $d = d_o$. Also by

lemma 3.4 $S_{c_o} \cap S_{d_o} = \emptyset$. This proves G1. If $s \in S_c \cap S_d$, then by G1 (already proved) either $c = c_o$, $d = d_g$ whence $(s, c_o, d, g(s)) \in M$ or $c = c_f$ and $d = d_o$ whence again $(s, c, d, f(s)) \in M$. G2 follows since

$$S_W \cup S_L = W \cup L \subseteq S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$$

by B3.

$$= S - \left(\left(\bigcup_{f \in F} S_f \right) \cup \left(\bigcup_{g \in G} S_g \right) \right)$$

$$= S - \bigcup_{c \in C} S_c = S - \bigcup_{d \in D} S_d$$

To show G3 one notes that $s \in S_{c_o} \cap S_d$ implies $s \in S_g$ where $d = d_g$. Hence by B4, $(c_o, d)(s) = g(s) \in S - W - \bigcup_{g \in G} S_g = S - S_W - S_{c_o}$. G4 follows similarly from B5.

It is also interesting to note that

<u>Theorem 3.6</u> If B is a board game and R a game-situation then

$$B = B(R(B))$$

$$\text{and } R = R(B(R))$$

<u>Proof</u>: Let $B = \langle S, F, G, W, L \rangle$

$R(B) = \langle S, C, D, M, S_W, S_L \rangle$

$B(R(B)) = \langle S, F', G', W, L \rangle$

Let $f \in F$ and $f(s) = s'$. Then there exists a $c_f \in C$ such that $(s, c_f, d_o, s') \in M$, i.e. $(c_f, d_o)(s) = s'$ whence $f_{c_f}(s) = s'$. Hence $f \subseteq f_{c_f}$. Again if $f_{c_f}(s) = s'$ then $(c_f, d_o)(s) = s'$, i.e. $(s, c_f, d_o, s') \in M$ which implies $f(s) = s'$ i.e. $f_{c_f} \subseteq f$ showing $f = f_{c_f}$. Hence $F \subseteq F'$.

Let now $f \in F'$ and $f = f_c$ where $c \in C$. If $f_c(s) = s'$ then $(s, c, d_o, s') \in M$ whence $c = c_{f'}$ for some $f' \in F$ such that $f'(s) = s'$ whence $f \subseteq f'$. Again if $f'(s) = s'$ then $(s, c_{f'}, d_o, s') = (s, c, d_o, s') \in M$ or $f_c(s) = s'$. That is $f' \subseteq f$ or $f' = f$. This shows $F' = F$.

The equality of G' and G can be proved analogously. This shows $B(R(B)) = B$.

Let now $R = \langle S, C, D, M, S_W, S_L \rangle$

$B(R) = \langle S, F, G, W, L \rangle$

$R(B(R)) = \langle S, C', D', M', S_W, S_L \rangle$

F has an unique element f_c for each element $c \in C$ where $c \neq c_o$ and C' has an unique element c_{f_c} for each element $f_c \in F$ and an additional element c_o. Hence the mapping $c \mapsto c_{f_c}$ between C and C' is one-one onto. So is the mapping $d \mapsto d_{g_d}$ from D to D'. Denoting d_{g_d} by d and c_{f_c} by c these maps may be considered as identity maps.

Let $(s, c, d, s') \in M$; by G1, either $c = c_o$ or $d = d_o$ but not both. Let $c = c_o$ then $g_d(s) = s'$. Hence $(s, c_o, d_g, s') \in M'$. Since $d_g = d$, $(s, c, d, s') \in M'$. Similarly if $d = d_o$, $(s, c, d, s') \in M'$. Hence $M \subseteq M'$.

Let now $(s, c, d, s') \in M'$. Either $c = c_o$ or $d = d_o$ by construction of R(B(R)) for B(R). If $c = c_o$, then $d = d_g$ for some $d \in D$ and $g_d(s) = s'$. From construction of B(R) for R this is true only if $(c_o, d)(s) = s'$, i.e. $(s, c_o, d, s') = (s, c, d, s') \in M$. Similarly if $d = d_o$, (s, c, d, s') M. Hence $M' \subseteq M$.

3. Winning Solutions in Board Games

Theorem 3.6 establishes the similarity of structure between Board games and game-situations. Most of the present chapter will deal with board games. In the rest of this section the concept of winning solution and winning strategies for board games will be introduced.

Given an element $s_o \in \bigcup_{f \in F} S_f$, a sequence $\mathcal{F} = (f_1, f_2, \ldots f_n;\ f_i \in F$ for all $1 \leq i \leq n)$, and a sequence $\mathcal{G} = (g_1, g_2, \ldots g_{n-1};\ g_i \in G$ for all i, $1 \leq i < n)$, $\mathcal{G}$ will be called compatible with $\mathcal{F}$ if and only if

$$f_1(s_o) \in S_{g_1};\ g_1(f_1(s_o)) \in S_{f_2};$$

$$f_2(g_1(f_1(s_o))) \in S_{g_2};\ g_2(f_2(g_1(f_1(s_o)))) \in S_{f_3}$$

....
....
....

for all $i < n-1$

$$f_{i+1}(g_i(f_i \ldots g_1(f_1(s_o)) \ldots) \in S_{g_{i+1}};$$

$$g_{i+1}(f_{i+1} \ldots g_1(f_1(s_o) \ldots)) \in S_{f_{i+2}}.$$

Given $s_o \in \bigcup_{f \in F} S_f$ and $\mathcal{F} = (f_1, \ldots f_2;\ f_i \in F$ for all i, $1 \leq i \leq n)$, will be called a winning solution for s_o if for each $\mathcal{G} = (g_1, g_2, \ldots g_{n-1};$ $g_i \in G$ for all i, $1 \leq i < n)$ compatible with $\mathcal{F}$

$$f_n(g_{n-1}(f_{n-1} \ldots g_1(f_1(s_o) \ldots) \in W.$$

As was indicated in Chapter I, the demand for a situation s_o to have a winning solution is an extremely restrictive one, corresponding to the demand for an open-loop control. The next few definitions introduce the less demanding ideas of winning strategies.

A function $Q_F: \bigcup_{f \in F} S_f \to F$ is called a board control strategy if for all s, $Q_F(s) = f$ implies $s \in S_f$. Similarly a function $Q_G: \bigcup_{g \in G} S_g \to G$ is called a board disturbance strategy if for all s, $Q_G(s) = g$ implies $s \in S_g$.

A board control strategy Q is called winning for $s_o \in S$ if there exists an integer N such that for every board disturbance strategy Q_G there exists sequences $(f_1, f_2, \ldots f_n;\ f_i \in F$ for all i, $1 \leq i \leq n)$ and $(g_1, g_2, \ldots g_{n-1};\ g_i \in G$ for all i, $1 \leq i < n)$ such that

(a) $n \leq N$

(b) $Q_F(s_o) = f_1;\ Q_G(f_1(s_o)) = g_1$

$Q_F(g_1(f_1(s_o))) = f_2;\ \ Q_G(f_2\ (g_1(f_1\ (s_o)))) = g_2$

for all $i < n-1$

$Q_F(g_i(f_i \ldots g_1(f_1(s_o)\ldots) = f_{i+1};\ Q_G(f_{i+1}(g_i \ldots (g_1(f_1(so)\ldots) = g_{i+1}$

$Q_F(g_{n-1}(f_{n-1} \ldots g_1(f_1(s_o)\ldots) = f_n$

and

(c) $f_n(g_{n-1}(f_{n-1} \ldots g_1(f_1(s_o)..) \in W$

For the sake of brevity (a) may be expressed by saying, "the sequences $\mathfrak{F}$ and $\mathfrak{G}$ have lengths n and n-1"; (b) may be expressed by saying, "the sequences $\mathfrak{F}$ and $\mathfrak{G}$ are dictated by Q_F and Q_G"; (c) may be expressed by saying, "the sequences $\mathfrak{F}$ and $\mathfrak{G}$ end s_o in W".

The reason for calling a demand for winning solutions stronger than a demand for winning steategies can be brought out by asking two questions:

i) If there is a winning solution for s_o, is there a winning board control strategy for s_o?

ii) If there is a winning board control strategy for s_o, is there a winning solution for s_o?

The answers to both these questions are "no" in general. In what follows a sufficient condition will be set forth for the answer to question 1 above to be "yes". (A necessary and sufficient condition can be developed with some effort, but is not worth doing.) A counter-example will indicate that the condition is not sufficient for the answer to the second question to be "yes".

A board game will be called <u>free</u> if for all f, f' $\in$ F and s $\in$ S $f_1(s) \in S_g$ implies $g(f(s)) \in S_{f'}$ or $g(f(s)) \in L$

<u>Theorem 3.7</u> If in a free board game there is a winning solution for s_o, there is a winning board control strategy for s_o.

<u>Proof</u>: Let $\mathfrak{F} = (f_1, f_2, \ldots f_n)$ be a winning solution for s_o. Let Q_G be a board disturbance strategy. Define a sequence $\mathcal{G}_{Q_G} = (g_1, g_2, \ldots g_{n-1})$ as follows:

$$g_1 = Q_G(f_1(s_o))$$

and for all i n

$$g_i = Q_G(f_i(g_{i-1}(f_{i-1} \ldots g_1(f_1(s_o) \ldots)$$

Q_G is compatible with $\mathfrak{F}$, since by definition of strategy

$$f_i(g_{2-1}(f_{i-1} \ldots g_1(f_1(s_o) \ldots) \ldots) \in S_{g_i}$$

and also, since the board game is free

$$g_{i-1}(f_{i-1} \ldots g_1(f_1(s_o) \ldots) \in S_{f_i}$$

Define a sequence of situations $T^{Q_G} = (s_o^{\mathcal{G} Q_G}, s^{\mathcal{G} Q_G} \ldots s_{n-1}^{\mathcal{G} Q_G})$ as follows

$$s_o^{\mathcal{G} Q_G} = s_o$$

and for each $i < n$

$$s_i^{\mathcal{G} Q_G} = g_i(f_{i-1}(s_{i-1}^{\mathcal{G} Q_G}))$$

With each element $s \in \bigcup_{f \in F} S_f$ associate a subset $F_s \subseteq F$ as follows:

$$F_s = \left\{ f_{i+1} \mid s = s_i^{\,\xi, Q_G} \text{ for some strategy } Q_G \right\}$$

Define Q_F: $\bigcup_{f \in F} S_f \to F$ as follows:

$$Q_F(s) = f_k$$

where k is the largest integer such that $f_k \in F_s$. $Q_F(s) = f_1$ if F_s is empty.

It will be shown that Q_F is a winning strategy for s_o. The proof is by induction on n, the number of components in $\mathcal{F}$.

If $n = 1$, $f_n(s_o) \in W$.

In this case $F_{s_o} = s_o$ whence $Q_F(s_o) = f_1$. Hence Q_F is a winning strategy for s_o.

Let now the theorem be true for $n < j$. Let $Q_F(s_o) = f_{k+1}$. If $k = 0$, then $(f_2, \ldots f_n)$ is a winning solution for $g_1(f_1(s_o))$ for all g_1 such that $f_1(s_o) \in S_{g_1}$, and hence Q_F is a winning strategy for all $g_1(f_1(s_o))$ such that $f_1(s_o) \in S_{g_1}$. If $k > 0$, then $(f_{k+2}, \ldots f_n)$ is a winning solution for all $g_{k+1}(f_{k+1}(s_o))$ such that $f_{K+1}(s_o) \in S_{g_{k+1}}$ and hence Q_F is a winning strategy for s_o.

The converse of this theorem is not true. Consider a board game as follows:

$S = (A,B,C,E,F,G,H,I,J,K,L)$

$L = \emptyset$

$W = (K,L)$

$F = (a,b,c)$

$G = (\alpha, \beta, \gamma)$

$S_a = S_b = S_c = (A,B,C,E,F)$

$S_\alpha = (G,H,I)$

$S_\beta = (G,J,I)$

$S_\gamma = (H,J)$

The functions $a,b,c,\alpha,\beta,\gamma$ are given in tabular form below and in graphic form in F.g. 3.1.

x	a(x)	b(x)	c(x)	y	(y)	(y)	(y)
A	I	G	J	G	C	A	-
B	I	J	H	H	E	-	F
C	H	I	J	I	A	B	-
E	J	K	I	J	-	C	B
F	L	J	I				

It can be seen by inspection that there is no winning solution for A. However, the strategy Q_F shown below is a winning strategy for A.

x	$Q_F(x)$
A	c
B	c
C	a
E	b
F	a

The concepts of strategy in board games and game situation are closely related and this enables one to indcate a theorem analogous to Theorem 2.1 for board games.

Given a control strategy P_C in a game situation R one defines a relation $B(P_C) \subseteq S \times F$ as follows:

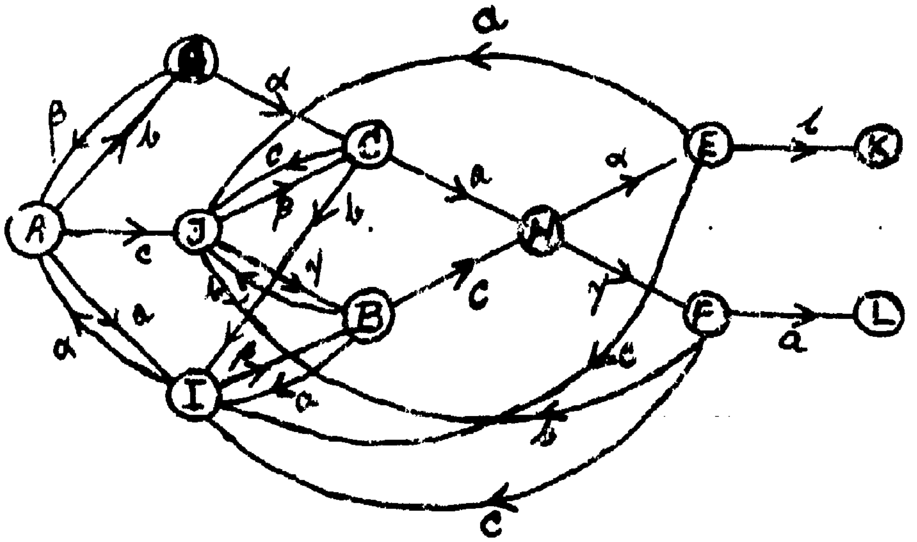


Fig. 3.1

$$(s,f) \in B(P_C)$$

if and only if $P_C(s) \neq c_o$ and $f = f_{P_C(s)}$. Similarly given a disturbance strategy P_D in a game situation R one defines a relation $B(P_D) \subseteq S \times G$ as follows:

$$(s,g) \in B(P_D)$$

if and only if $P_D(s) \neq d_o$ and $g = g_{P_D(s)}$.

The following lemmata can be proved readily:

<u>Lemma 3.8</u> If P_C and P_D are control and disturbance strategies in a game situation R then

i) $B(P_C)$ is a board control strategy in $B(R)$

ii) $B(P_D)$ is a board disturbance strategy in $B(R)$

<u>Proof</u>: Only (i) will be proved. The proof of (ii) follows identically.

To show that $B(P_C)$ is a function $B(P_C): {}_{f\ F}S_f \quad F$ let $(s,f) \in B(P_C)$, then $P_C(s) \neq c_o$, whence $s \in \bigcup_{c \neq c_o} S_c = \bigcup_{f \in F} S_f$ whence the domain of $B(P_C)$ is contained in $\bigcup_{f \in F} S_f$. Again, if $s \in \bigcup_{f \in F} S_f$, then $s \in \bigcup_{c \neq c_o} S_c \subseteq \bigcup_{c \in C} S_c$. Hence $P_C(s)$ is defined and $P_C(s) \neq c_o$. If $P_C(s) = c$, then $(s, f_c) \in B(P_C)$. Hence $\bigcup_{f \in F} S_f$ is contained in the domain of $B(P_C)$. Hence the domain of $B(P_C)$ coincides with $\bigcup_{f \in F} S_f$. If $(s,f) \in B(P_C)$ and $(s,f') \in B(P_C)$ then $f' = f_c$ and $f = f_c$ by definition of $B(R)$. Since f_c is unique $f = f'$ showing that $B(P_C)$ is a function.

Let now $B(P_C)(s) = f$, then $P_C(s) = c$ where $f = f_c$. Since $s \in S_c$ by definition of control strategy and $S_c = S_{f_c}$, $s \in S_f$, fulfilling the condition for $B(P_C)$ being a board control strategy.

<u>Lemma 3.9</u> i) The mapping $P_C \mapsto B(P_C)$ is a one-one map onto the set of all board control strategies of $B(R)$. Similarly (ii) the mapping $P_D \mapsto B(P_D)$ is a one-one map onto the set of all board disturbance strategies of $B(R)$.

<u>Proof</u>: Let P_C and P'_C be two distinct control strategies, so that for at least one $s \in \bigcup_{c \in C} S_c$, $P_C(s) \neq P'_C(s)$. Since $P_C(s) = c_o$ for all $s \in S_{c_o}$; this implies $s \notin S_{c_o}$. Hence both $B(P_C)(s)$ and $B(P_C)(s')$ are defined. Let $P_C(s) = c \neq c' = P'_C(s)$. By definition of $B(P_C)$ and $B(P'_C)$.

$$B(P_C)(s) = f_c;\ B(P'_C)(s) = f_{c'}.$$

But the map $c \mapsto f_c$ is one-one by definition, hence $c \neq c'$ implies $f_c \neq f_{c'}$. Hence $B(P_C) \neq B(P_{C'})$.

To show that every board control strategy Q_F is equal to $B(P_C)$ for some control strategy P_C, one constructs the control strategy as follows:

$$P_C(s) = \begin{cases} \text{(i)} & c \text{ if and only if } Q_F(s) = f_c \\ \text{(ii)} & c_o \text{ if and only if } s \in \bigcup_{g \in G} S_g \\ \text{(iii)} & \text{undefined otherwise} \end{cases}$$

P_C is a control strategy, since $P_C(s)$ is defined for $\bigcup_{f \in F} S_f = \bigcup_{c \neq c_o} S_c$ by construction (i) and over S_{c_o} by condition (ii). Also if $P_C(s) = c$ then $Q_F(s) = f_c$. $Q_F(s) = f_c$ indicates $s \in S_{f_c} = S_c$. Also $P_C(s) = c_o$ only if $s \in S_{c_o}$. Again $Q_F = B(P_C)$ by construction.

The most important thing to notice about the mapping $P_C \mapsto B(P_C)$ is given by the next theorem.

<u>Theorem 3.10</u> In a game situation R, P_C is a winiing control strategy for $s_o \in \bigcup_{c \neq c_o} S_c$ if and only if $B(P_C)$ is a winning board control strategy for s_o in B(R).

<u>Proof</u>: Let P_C be a winning control strategy for s. Then given any disturbance strategy P_D, there exists a sequence $(c_1,d_1)\ldots(c_n,d_n)$ such that

$$c_1 = P_C(s_o);\quad d_1 = P_D(s_o)$$

for each $i < n$

$$c_{i+1} = P_C((c_i, d_i)((c_{i-1}, d_{i-1})(\ldots(c_1, d_1)(s_o))\ldots)$$

$$d_{i+1} = P_C((c_i, d_i)((c_{i-1}, d_{i-1})(\ldots(c_1, d_1)(s_o))\ldots)$$

and

$$(c_n, d_n)((c_{n-1}, d_{n-1})(\ldots(c_1, d_1)(s_o))\ldots) \in W$$

Since $s_o \in \bigcup_{c \neq c_o} S_c$ $c_1 \neq c_o$. Also, $d_1 = d_o$. Also, since

$$(c_n, d_n)((c_{n-1}, d_{n-1})(\ldots(c_1, d_1)(s_o))\ldots) \notin W,$$

one obtains from the contrapositive of G3, that

$$((c_{n-1}, d_{n-1})\ldots(c_1, d_1)(s_o))\ldots) \in S_{c_o} \cap S_d$$

for some $d \in D$. Hence $d_n = d_o$. Hence,

$$c_1 = c_3 = \ldots c_n \neq c_o, \; c_2 = c_4 = \ldots c_{n-1} = c_o$$

$$d_1 = d_3 = \ldots d_n = d_o, \; d_2 = d_4 = \ldots d_{n-1} \neq d_o$$

indicating that n is an odd integer; let $n = 2m_o - 1$.

Set $f_{c_{2m-1}} = f_m$ for each $m \leq m_o$ and $g_{d_{2m}} = g_m$ for each $m < m_o$. By definition of B(R) for each $m \leq m_o$ $f_m = (c_{2m-1}, d_o) = (c_{2m-1}, d_{2m-1})$ and for each $m < m_o$

$$g_m = (c_o, d_{2m}) = (c_{2m}, d_{2m})$$

reducing the equation

$$(c_n, d_n)((c_{n-1}, d_{n-1})(\ldots(c_1, d_1)(s_o)\ldots) \in W$$

$$f_{m_o}(g_{m_o-1}(\ldots g_1(f_1(s_o))\ldots) \in W.$$

Also, for each even i, $c_{i+1} = P_C((c_{i-1}, d_{i-1})(\ldots(c_1, d_1)(s_o))\ldots)$

reduces to

$$f_{(i/2)+1} = P_C(g_{i/2}(f_{i/2-1}\ldots g_1(f_1(s_o))\ldots))$$

and for each odd i

$$d_{i+1} = P_D((c_i, d_i)(\ldots(c_1, d)(s_o))\ldots)$$

reduces to

$$g_{(i+1)/2} = P_C(f_{i-1/2}(g_{i-1/2}(\dots g_1(f_1(s_o))\dots).$$

However, by definition of $B(P_C)$

$$P_C(s) = c_{i+1}$$

if and only if $B(P_C)(s) = f_{(i/2)+1}$ since $c_{i+1} \neq c_o$. Similarly

$$P_D(s) = d_{i+1}$$

if and only if $B(P_D)(s) = g_{(i+1)/2}$. Hence one obtains

$$f_1 = Q_F(s_o) = f_1; \quad Q_G(f_1(s_o)) = g_1$$

and for all $i \quad n+1/2 = m_o$

$$B(P_C)(g_i(f_i(\dots g_1(f_1(s_o)\dots) = f_{i+1}$$

$$B(P_D)(f_{i+1}(g_i\dots g_1(f_1(s_o)\dots) = g_{i+1}$$

and

$$B(P_C)(g_{m_o-1}(f_{m_o-1}\dots g_1(f_1(s_o)\dots) = f_{m_o}$$

Since P_C is a winning strategy, $B(P_C)$ is such that for any board-disturbance strategy $B(P_D)$ a sequence $f_1, f_2 \dots f_{m_o}$ and a sequence $g_1, g_2 \dots g_{m_o-1}$ will exist having the above properties $B(P_C)$, hence, is a winning strategy.

The proof of the "if" part of the theorem is left to the reader. The theorem leads to the following interesting corollary analogous to a weak form of theorem 2.1.

<u>Corollary 3.11</u> In a board game B, there exists a board control strategy which is winning for every element $s \in \bigcup_{f \in F} S_f$ such that a winning strategy for s exists.

<u>Proof</u>: If $s \in \bigcup_{f \in F} S_f$ in B then $s \in \bigcup_{c \neq c_o} S_c$ in R(B). If a winning board control strategy exists for s in B then by theorem 3.10 a winning control strategy exists for s in R(B). Hence the set of all $s \in \bigcup_{f \in F} S_f$ for which a winning board control strategy exists is a subset of the set of all $s \in S$ for which a winning control strategy exists. Hence it follows a forteriori from Theorem 2.1 that there exists a strategy P_C in R(B) which is

a winning control strategy for all elements of this set. a second application of Theorem 3.10, $B(P_C)$ will be a winning strategy for all the elements of this set.

It will be noted that Corollary 3.11 neglects to make any statements regarding non-losing strategies. It seems apparent that a stronger form for Corollary 3.11 could be obtained. However, since most of the later discussion will be directed towards winning board control strategies for members of $\bigcup_{f \in F} S_f$ in board games, extensions to such stronger forms may not be relevant at present.

A discussion regarding strategies and their descriptions similar to that in sec. 6 of Chapter II is pertinent here. Corresponding to each strategy Q_F, $Q_F \cdot Q_F^{-1}$ again defines a partition of $\bigcup_{f \in F} S_f$. As pointed out before, the major problem regarding the applicability of any winning strategy (even when it is definable) lies in the ease of describing the elements of the partition which its Kernel induces. Also, if one is interested in only a small subset of situations, one has a greater freedom of choosing between alternative strategies of varying ease of applicability.

The major problem regarding winning strategies, of course, remains: "How does one find a winning strategy?" As in Chapter II, the later sections of the present chapter will deal with certain aspects of this problem as applied to board games. Initially, however, a few well-known games will be described in the general format of board games. This also will be in keeping with what has been done in Chapter II.

4. The NIM Class of Games - An Example

This and the next section will describe two classes of well-known games. As in Chapter I, the examples serve to illustrate the suitability of the formal model of board games for well-known cases and provide vehicles for discussion in later chapters.

The first class of games can be described in general terms as follows. One has a number of piles of sticks on the table. Each player in his turn removes a number of sticks from each pile obeying certain restrictions (for instance, "not more than one stick from each pile," "sticks to be removed only from one pile" and such like). The first player to pick the last stick in the pile wins. (In some variants of the games the person who takes the last stick loses: but the difference is not essential and in this book the rule will be as stated initially). Specific games in this class will be distinguished by the number of piles, the number of sticks on each pile initially and (in a more fundamental way) by the constraints on the way the sticks can be removed by each player.

The set S of situations in all of these games is characterized by a set of ordered pairs (I,p) p determines which player is to move (this will be formally stated presently). I is a sequence of n non-negative integers, where n is the number of piles and each integer in the sequence denotes the number of sticks in each pile. Any situation $s \in S$, then, has the form $((i_1,i_2,\dots i_n),p)$ where i_k is an integer for each k ($1 \leq k \leq n$) and p is either the integer 0 or the integer 1.

Each element of the set of functions F has the form (x,0) where x is a sequence of n non-negative integers $x = (x_1,x_2,\dots x_n)$. Unlike the sequence I, however, where any sequence of integers is permitted, x has to satisfy some criterion according to the rules of the game. We shall specify this criterion in general by a statement which x must satisfy i.e. such that $\alpha(x)$ is true for any x such that (x,0) is a member of F.

Similarly, each element of the set of functions G has the form $(x,1)$ where x is a sequence of n non-negative integers $x = (x_1, x_2, \ldots x_n)$ satisfying some criterion β.

For any element $(x,p) \in F \cup G$, $S_{(x,p)}$ is defined as follows

$$S_{(x,p)} = \{ (I,p) \mid i_k \geq x_k \text{ for each } k(1 \leq k \leq n)$$

and for each $(I,p) \in S_{(x,p)}$

$$(x,p)((I,p)) = ((i_1', i_2', \ldots i_n'), \ p + 1 (\text{mod. } 2))$$

where for each $k(1 \leq k \leq n)$

$$i_k' = i_k - x_k.$$

W consists of the single element $((0,0,\ldots 0),1)$ and L of the single element $((0,0,\ldots 0),0)$. α and β are so chosen that B3 is always satisfied. It is left to the reader to verify that B1, B2, B4 and B5 are satisfied by any specification in the class defined above.

The description of some specific games follow.

The simplest sub-class of games in this class occurs when $n=1$, $\alpha \equiv \beta \equiv (x_1 \leq k)$ with a specific k. A typical game of this class, may be "There are 15 sticks in a pile. Each player in his turn takes away at least 1 and at most 3 sticks from the board. The player who leaves an empty pile wins". Here the initial state is taken to be $((15),0)$ or $((15),1)$ depending on who plays first.

A specific game in the larger class which is easy to analyze is one in which $n=2$ and $\alpha \equiv \beta \equiv ((x_1 \leq 1) \text{ and } (x_2 \leq 1) \text{ and } (x_1 + x_2 \neq 0))$. With the initial state $((5,3),0)$ the game is described as follows, "There are two piles, with 5 and 3 sticks. Each player, in his turn, picks up at least one stick, but not more than 1 from each pile. The player who leaves both piles empty wins".

In one of the most well-known sub-class of this class of games

$$\alpha * \beta \equiv (\exists x_i)(x_i > 0 \text{ and } j \neq i \rightarrow x_j = 0)$$ that is sticks are removed from one and only one pile. The well-known game of Nim belongs to this class; in this specific game n=3, and the initial state is ((3,5,7),0) or ((3,5,7),1) depending on who plays first. It will also be of interest to consider a more general sub-class of this class of games where

$$\alpha * \beta \equiv (\exists x_i)(k \geq x_i > 0 \text{ and } j \neq i \rightarrow x_j = 0).$$

These games will be referred to as various methods for finding strategies are developed in later sections.

5. The Tic-Tac-Toe-Like Games - Another Example

The class of games to be discussed in this section are of interest in this book in view of the fact that a close examination of these bring out in a convincing and non-trivial way the close relationship that exists between the efficiencies of solution and description languages. The class of languages will be described here without any reference to the description language. The significance of the description languages to this class of games will be discussed in a later chapter. A few well-known members of this class will then be exhibited.

All the games in this class can be visualized as played on a board consisting of a finite number of "cells". Two classes of subsets of the set of all cells are pre-defined, which we shall call $\mathcal{A}$ and $\mathcal{B}$. The members of $\mathcal{A}$ will be denoted by A (with or without subscripts) and will be called "winning files for x"; members of $\mathcal{B}$ will be denoted by B with or without subscripts and called "winning files for Y".

In the beginning, each cell is unmarked. The players play alternately. The first player, in his turn, marks some previously unmarked cell with an "X"; the second player, in his turn, marks some previously unmarked cell with a "Y". The first player wins if, on making his mark, a configuration of marks is produced such that some winning file for X has an "X" on each of its cells. The second player wins if, on making his mark, a configuration is produced such that some winning file for Y has "Y" on each of its cells.

Formally, with each game will be associated a finite set N, and two class $\mathcal{A}$ and $\mathcal{B}$ such that

$$A \in \mathcal{A} \text{ implies } A \subset N$$

$$B \in \mathcal{B} \text{ implies } B \subset N$$

One can assume without losing any essential aspect of the games, that no member of $\mathcal{A}$ is a proper subset of any other member of $\mathcal{A}$; and similarly for $\mathcal{B}$. Another set with three elements $\{X, Y, \Lambda\}$ will also be used in specifying any game in the class.

Given $N, \mathcal{A}, \mathcal{B}$ for a game, one defines a board game as follows:

Any situation s is a function from N into $\{X, Y, \Lambda\}$ such that the number of cells mapped into X is equal or one more than the number of cells mapped into Y. Denoting the cardinality of set P by $|P|$, one may say the above formally as follows

$s \in S$ if and only if $s \in \{X, Y, \Lambda\}^N$ and $((|s^{-1}(X)| = |x^{-1}(Y)|)$ or $(|s^{-1}(X)| = |x^{-1}(Y)| + 1))$

$s \in W$ if and only if $|s^{-1}(X)| = |s^{-1}(Y) + 1|$, there exists an unique file $A \in \mathcal{A}$ such that $A \subseteq s^{-1}(X)$ and there is no file $B \in \mathcal{B}$ such that $B \subseteq s^{-1}(Y)$.

$s \in L$ if and only if $|s^{-1}(X)| = |s^{-1}(Y)|$, there exists an unique file $B \in \mathcal{B}$ such that $B \subseteq s^{-1}(Y)$ and there is no file $A \in \mathcal{A}$ such that $A \subseteq s^{-1}(X)$.

Each element of F is denoted by the pair (n,X) where n is an element of N. Every element of G is denoted by a pair (n,Y) where n is an element of N.

$s \in S_{(n,X)}$ if and only if $s \in S-L$, $|s^{-1}(X)| = |s^{-1}(Y)|$, and $s(n) = \Lambda$. In this case $(n,X)(s) = s'$ where $s'(m) = s(m)$ if $m \neq n$ and $s'(n) = X$.

$s \in S_{(n,Y)}$ if and only if $s \in S-W$, $|s^{-1}(X)| = |s^{-1}(Y)| + 1$ and $s(n) = \Lambda$. In this case $(n,Y)(s) = s'$ where $s'(m) = s(m)$ if $m \neq n$ amd $s'(n) = Y$.

As in the last section, it will be left to the reader to verify that B1) to B5) are satisfied by any board game defined as above. In what follows some well-known games in this class will be described.

The most well-known sub-class of this class of games is the "m^n Tic-Tac-Toe" games. 3^2 Tic-Tac-Toe or "Naughts and Crosses" is the most popular one among young children. 4^3 Tic-Tac-Toe is a game sophisticated enought to be played by adults and sells under the trade name, "Qubic".

In a general m^n tic-tac-toe game the set N consists of n-tuples of integers each element of the n-tuple being a non-negative integer less than m. S, then is a pre-specified subset of $\{X,Y,\Lambda\}^{m^n}$.

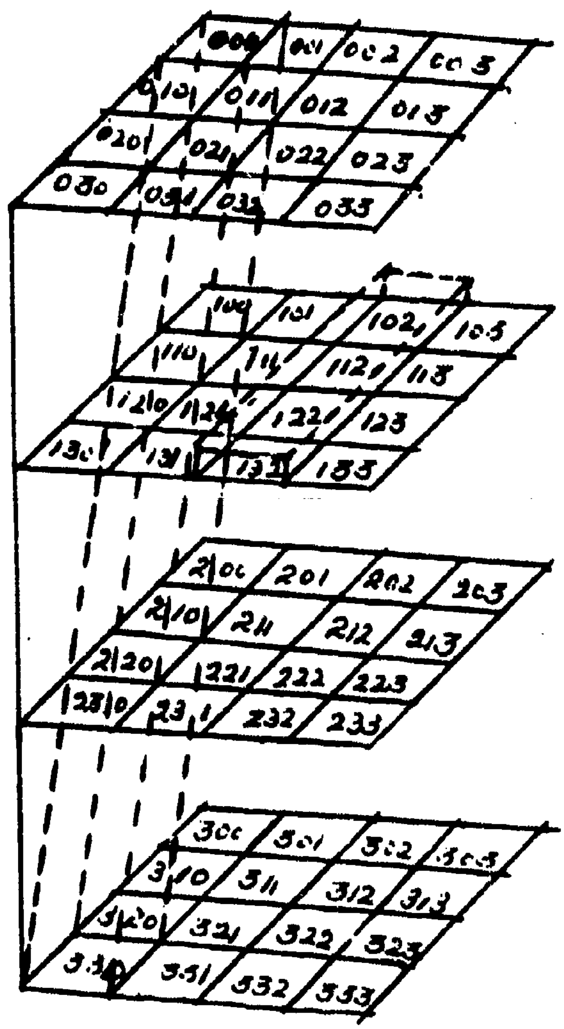
The classes $\mathcal{A}$ and $\mathcal{B}$ coincide in this class of games and consist in the set of n typles defined as follows

$$A \in \mathcal{A} \equiv \quad A = \left\{ (f_1(s), f_2(s), \ldots f_n(s)) \,\middle|\, 0 \leqslant s < m \right\}$$

and each $f_i(s)$ is either a constant between 0 and m-1 inclusive or $f_i(s) = s$ or $f_i(s) = m-1-s$: but not all f_i can be constant functions.

Basically, the above formalism states that a set is a file if it consists of m cells in a straight line. The idea can be exemplified by exhibiting the picture of a Qubic (4x4x4 tic-tac-toe) board and two lines on it, as shown by the shaded parallelopiped. One consists of the 4 cells $\{(0,0,0), (1,1,0), (2,2,0), (3,3,0)\}$ which can be represented by $\{(s,s,0) \mid 0 \leqslant s < 4\}$. The other consists of the cells $\{(1,0,2), (1,1,2), (1,2,2), (1,3,2)\}$ which can be represented by $\{(1,s,2) \mid 0 \leqslant s < 4\}$. The set represented by $\{(s,2,3-s) \mid 0 \leqslant s < 4\}$ consists of the cells $\{(0,2,3), (1,2,2), (2,2,1), (3,2,0)\}$.

The files in m^n tic-tac-toe are easier to describe intuitively for small values of n. However, something like the formal description given above (which is just a parametric definition of straight lines in a "lattice") is essential for machine representation. This particular representation of files as n-tuples of functions has been found useful in certain combinatorial problems associated with multiplicity of various classes of files in general m^n tic-tac-toe games.



The 4^3 Tic-Tac-Toe Game

Fig. 3.7

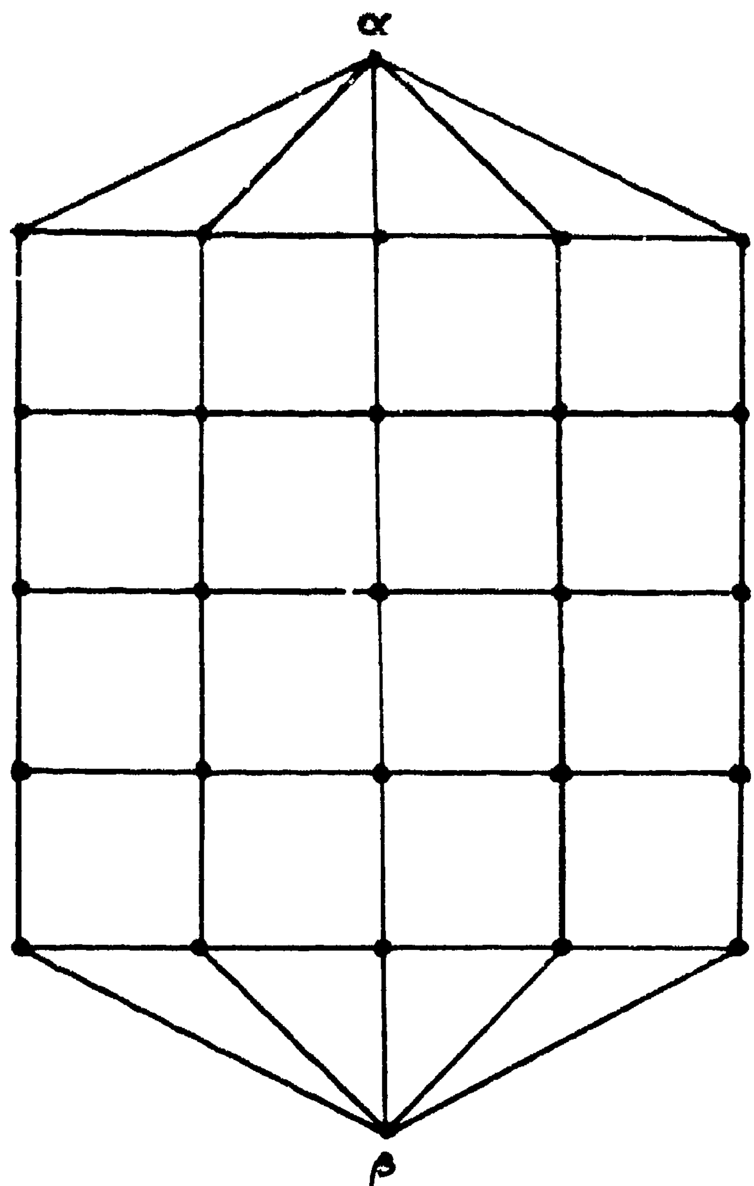
Another well-known game in the Tic-Tac-Toe-like class is Go-Moku (also known as Renjyu, Pegetty and 5-place Tic-Tac-Toe). The set N consists of cells in a 19x19 board as in 19^2 tic-tac-toe game. However, the files, instead of being sets of 19 elements, are sets of 5 elements in a line anywhere on the board. Thus, the files in $\mathcal{A}$ and $\mathcal{B}$ consist of sets of the form $\left\{(f_1(s),f_2(s)) \mid 0 < s \leq 5\right\}$ where f_1 and f_2 have the form K, K + S, K - S where K is any non-negative integer less than 16 and more than 4: it being specified that both f_1 and f_2 are not constant functions.

In a third class of games the set N consists of arcs in a specified graph with two designated nodes. The class $\mathcal{A}$ consists of all paths between the designated points and $\mathcal{B}$ the class of all minimal sets of arcs whose removal separates the designated points. This is often described by saying that the first player, in his turn renders an arc invulnerable while the second player, in his turn, removes one of the invulnerable arcs. The game continues till either an invulnerable path is established between the two designated nodes or the nodes have been separated. In the first case the first player wins. In the second case the second player wins.

Games in this class are called Shannon games after their originator. Lehman [21] has recently given a characterization of the class of networks for which there is a winning strategy for the initial configuration. The strategy given by him is characterized differently from the general strategy for Tic-Tac-Toe-like games discussed later in this book.

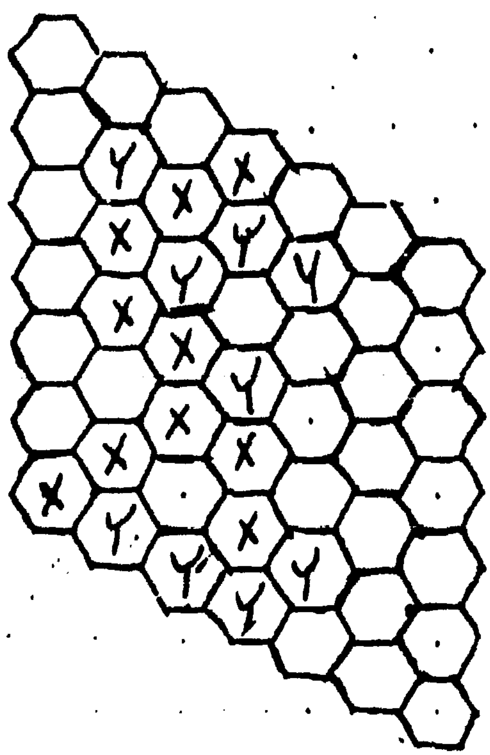
The most important difference between the Shannon games and the m^n Tic-Tac-Toe games lies in the fact that the classes $\mathcal{A}$ and $\mathcal{B}$ are described in a much more complicated way. This has a rather strong import on the way these games are played.

A specific Shannon game is commercially available under the name, "Bridg-it". It consists of the network shown in Fig. 3.3, with the nodes



The Shannon Game For Bridg-it

Fig. 3.3



A Winning Solution in 7 X 7 Hex

Fig. 3.4

α and β pre-designated. The actual bridge-it board, and the way the game is described, appears different from the above consideration. However, Busacker and Saaty [22] have pointed out that if α and β are joined by an extra arc, then the dual of the resulting graph is isomorphic to the resulting graph and the dual of the cut sets of this graph are paths between two specific points of the dual graph (corresponding to the regions separated by the extra line) with the arc dual to the extra line removed. In the commercial game, the two opposing players play on the two dual graphs.

Another important game in the larger class of Tic-Tac-Toe like games is the game of Hex. The set N consists of hexagons on a honeycomb structure as shown in Fig. 3.4. The class $\mathcal{A}$ has as members all paths connecting the top-edge to the bottom edge. The class $\mathcal{B}$ consists of all paths connecting the left edge to the right edge. The figure exhibits a winning position.

Before leaving the subject it may be worthwhile to point out that every game in this class can be considered as a sub-game of a larger game in another class. A description of this class will be introduced here. This embedding will bring out certain essential symmetries between the control and disturbance which will be of interest in a later section.

The specification starts with the same triple, N, $\mathcal{A}$ and $\mathcal{B}$; the situations are ordered pairs $\{s,p\}$ where $p = 0$ or $p = 1$ and $s \in \{X,Y,\Lambda\}^N$ without any restriction on s. Elements of F and G again have forms (n,X) and (n,Y); however their definitions will be changed slightly as follows:

$(s,p) \in S_{(n,X)}$ if and only if $s \in S-L$, $s(n) = \Lambda$ and $p = 0$. In this case $(n,X)(s,0) = (s',1)$ where $s'(m) = s(m)$ if $m \neq n$ and $s'(n) = X$.

$(s,p) \in S_{(n,Y)}$ if and only if s S-W, s(n) = and $p = 1$. In this case $(n,Y)(s,1) = (s',0)$ where $s'(m) = s(m)$ if $m \neq n$ and $s'(n) = Y$.

Also, W and L are slightly re-defined as follows: $(s,p) \in W$ if and only if $p = 1$, there exists an unique file $A \in \mathcal{A}$ such that $A \subseteq s^{-1}(X)$; and there is no file $B \in \mathcal{B}$ such that $B \subseteq s^{-1}(Y)$; $(s,p) \in L$ if and only if $p = 0$, there exists an unique file $A \in \mathcal{B}$ such that $A \subseteq s^{-1}(Y)$ and there is no file B such that $B \subseteq s^{-1}(X)$.

The reader should convince himself that in the original version of the game the situations were restricted to the set of first components of all situations (s,p) of the new version which could be attained from $((\Lambda, \Lambda, \ldots \Lambda), 0)$.

In a later chapter certain subsets of S associated with winning strateiges of Tic-Tac-Toe-like games will be pointed out and the merits and drawbacks of the resulting strategies will be discussed.

6. Evaluating Strategies in Board Games

The practical utility of a winning control strategy is intimately related with the ease with which the blocks of its Kernel is amenable to easy description. Any comment that can be made regarding this matter has already been made in Section 6 of the previous chapter. It is therefore germaine to move directly to the discussion of various methods for finding winning strategies. The discussion in this section will roughly follow the same lines as Sections 7 and 8 of the previous chapters. However, due to the important role played by the disturbance in a board game, there will be more to say about evaluations in board games.

Given a board game one can define readily two classes of sets $\{W_i \mid i>0\}$ and $\{L_i \mid i>0\}$ as follows:

$s \in W_1$ if and only if there exists an $f \in F$ such that $f(s) \in W$.

$s \in W_{i+1}$ if and only if $s \notin W_k (k \leq i)$ and there exists an $f \in F$ such that $s \in S_f, f(s) \in \bigcup_{g \in G} S_g$ and for every $g \in G$ such that $f(s) \in S_g$, $g\ (f(s)) \in W_k (k \leq i)$.

$s \in L_1$ if and only if there exists $g \in G$ such that $g(s) \in L$.

$s \in L_{i+1}$ if and only if $s \notin L_k (k \leq i)$ and there exists a $g \in G$ such that $s \in S_g g(s) \in \bigcup_{f \in F} S_f$ and for every $f \in F$ such that $g(s) \in S_f$, $f(g(s)) \in L_k\ (k \leq i)$.

It is clear that $\bigcup_{k \geq 1} W_k \subseteq \bigcup_{f \in F} S_f$ and $\bigcup_{k \geq 1} L_k \subseteq \bigcup_{g \in G} S_g$. The following also can be shown readily.

<u>Theorem 3.6</u> i) If $s \in \bigcup_{k \geq 1} W_k$, then there exists an $f \in F$ such that $s \in S_f$ and $f(s) \notin \bigcup_{k \geq 1} L_k$.

ii) If $s \in \bigcup_{k \geq 1} L_k$, then there exists $g \in G$ such that $s \in S_g$ and $g(s) \notin \bigcup_{k \geq 1} W_k$.

Proof: Let $s \in W_k \subseteq \bigcup_{k\geq 1} W_k$. The proof will be by induction over k.

If $s \in W_1$, then there exists an $f \in F$ such that $f(s) \in W \subsetneq S - \bigcup_{g\in G} S_g$. Since $f(s) \notin \bigcup_{g\in G} S_g$ and $\bigcup_{k\geq 1} L_k \subseteq \bigcup_{g\in G} S_g$, $f(s) \notin \bigcup_{k\geq 1} L_k$.

Let the theorem be true for $s \in W_i$ $(i \leq k)$. Let $s \in W_{k+1}$. Then there exists an $f \in F$ such that $f(s) \in \bigcup_{g\in G} S_g$ and for all g such that $s \in S_g$, $g(f(s)) \in W_i (i \leq k)$. If $f(s) \in \bigcup_{k\geq 1} L_k$, let $f(s) \in L_p$. Then there exists a g such that $g(f(s)) \in \bigcup_{f\in F} S_f$ and for all f' such that $g(f(s)) \in S_{f'}$, $f'(g(f(s))) \in \bigcup_{k\geq 1} L_k$. However $g(f(s)) \in W_i$ $(i \leq k)$, whence there exists an $f' \in F$ such that $s \in S_{f'}$, and $f'(g(f(s))) \notin \bigcup_{k\geq 1} L_k$. This leads to a contradiction.

ii) can be proved similarly.

$\{L_i\}$ and $\{W_i\}$ will be called W-evaluations and L-evaluations respectively. This idea of evaluations follows the mode set by Chapter II. However, because the structure of board games is richer than that of W-problems, some further classes of sets related to evaluations can be utilized for the construction of strategies. Before taking up such further structures in detail, however, the results analogous to those in Chapter II will be set down first. Theorems analogous of Theorems 2.4 and 2.5 will be seen to hold true for board games, again as special cases of Theorem 2.1. In view of the more general structure of board games as compared to W-problems, it will be more meaningful and easier to prove these analogs in somewhat stronger forms. For this, a few more initial definitions are in order.

Given a board game $\langle S,G,F,W,L\rangle$ and an element $s_o \in \bigcup_{f\in F} S_f$ such that a winning board control strategy Q_F exists for s_o, one has a positive integer N such that for every board disturbance strategy Q_G there exists sequences $(f_1,f_2,\ldots f_n \mid f_i \in F)$ and $(g_1,g_2,\ldots g_{n-1} \mid g_i \in G)$ which fulfills the condition set out in Section 3. The integer n (less than or equal to N) is determined by s_o, Q_F and Q_G and will be denoted by $n(s_o,Q_F,Q_G)$ to emphasize this dependence. In view of Q_F being a winning strategy, $n(s_o,Q_F,Q_G) \leq N$ for every disturbance strategy Q_G. Hence a least upper

bound $n(s_o,Q_F)$ exists for the set of integers $\left\{ n(s_o,Q_F,Q_G) \mid Q_G \text{ is a board disturbance strategy} \right\}$. In symbols

$$n(s_o,Q_F) = \underset{Q_G}{\text{l.u.b.}} \left\{ n(s_o,Q_F,Q_G) \right\} \leqslant N$$

Since the set of integers $n(s_o,Q_F,Q_G)$ is finite, this bound is attained by some Q_G.

The greatest lower bound of $n(s_o,Q_F)$ over all winning strategies for s_o will be denoted by $n(s_o)$.

$$n(s_o) = \underset{Q_F}{\text{g.l.b.}} \left\{ n(s_o,Q_F) \mid Q_F \text{ is a winning strategy for } s_o \right\}.$$

Again, since the set $n(s_o,Q_F)$ is finite, this bound is attained by some Q_F.

The following lemma will be useful.

Lemma 3.7: In a board game, let there be a winning strategy for $s_o \in \bigcup_{f \in F} S_f$ and let $n(s_o) = n(s_o,Q_F) = n(s_o,Q_F,Q_G)$ for some control strategy Q_F and disturbance strategy Q_G. Let $s_1 = Q_F(s_o)(s_o)$ and $s' = Q_G(s_1)(s_1)$.

Then $n(s') \leqslant n(s_o) - 1$.

Proof: It can be seen initially that Q_F is a winning strategy for s'. If it is not a winning strategy, there exists a disturbance strategy Q_G' such that there are no sequence $\bar{f}$ and $\bar{g}$ of length $n(s')$ and $n(s')-1$ dictated by Q_F and Q_G' which end s' in W. If one now defines a new strategy Q_G'' such that $Q_G''(s_1) = Q_G(s_1)$ and $Q_G''(s) = Q_G'(s)$ for all situations $s (\neq s_1)$ for which Q_G' is defined, then there will be no $\bar{f}$, $\bar{g}$ of lengths $n(s_o)$ and $n(s_o)-1$ dictated by Q_F and Q_G'' which end s_o in W. This contradicts the hypothese that $n(s_o,Q_F) = n(s_o)$.

Given that there is a winning strategy for s', if $n(s') \lessdot n(s_o)-1$ then for every winning control strategy Q_F' $n(s',Q_F') \geqslant n(s_o)$. Hence there is a disturbance

strategy Q_G' such that $n(s',Q_F',Q_G') \geq n(s_o)$. A forteriori there is a disturbance strategy Q_G' such that $n(s',Q_F,Q_G') \geq n(s_o)$. Define a strategy Q_G'' such that $Q_G''(s_1) = Q_G(s_1)$ and $Q_G''(s) = Q_G'(s)$ for all $s(\neq s_1)$ for which Q_G' is defined. Then $n(s_o,Q_F,Q_G'') \geq n(s_o) + 1$. But $n(s_o) = n(s_o,Q_F,Q_G) \leq n(s_o,Q_F,Q_G'')$, leading to a contradiction.

One can easily prove the following on the basis of this.

Lemma 3.8 If in a board game $n(s_o) = k$, then $s_o \in \bigcup_{i=1}^{k} W_i$.

Proof: Let $n(s_o) = 1$. Then there exists a function $f \in F$ such that $f(s) \in W$. Hence $s_o \in W_1 = \bigcup_{i=1}^{1} W_i$.

Let now the theorem be true for $n(s_o) = k$. Let $n(s_o) = k + 1$. Then there exist a control strategy Q_F, such that for every disturbance strategy Q_G $n(s_o,Q_F,Q_G) \leq k + 1$. Let $Q_F(s_o)(s_o) = s_1$ and $Q_G(s_1)(s_1) = s'$. Then by lemma 3.7 $n(s') = k_o \leq k$. Hence by the induction hypothesis $s' \in \bigcup_{i=1}^{k_o} W_i \subseteq \bigcup_{i=1}^{k} W_i$. Hence by definition $s_o \in \bigcup_{i=1}^{k+1} W_i$.

This leads immediately to the following Corollary.

Corollary 3.9 If there is a winning control strategy for $s_o \in \bigcup_{f \in F} S_f$ then $s_o \in \bigcup_i W_i$.

Proof: Let $n(s_o) = k$. Then $s_o \in \bigcup_{i=1}^{k} W_i \subseteq \bigcup_i W_i$.

Analogous to the case of W-problems the idea of W-evaluations is of utility in the description of strategies. A strategy Q_F will be called evaluating if $s \in W_k$ $(k>1)$ and $Q_F(s) = f$ implies that for all $g \in G$ such that $f(s) \in S_g$, $g(f(s)) \in \bigcup_{k=1}^{k-1} W_i$, and $s \in W_1$ and $Q_F(s) = f$ implies $f(s) \in W$.

Theorem 3.10 An evaluating strategy is a winning strategy for every $s \in \bigcup_{f \in F} S_f$ for which a winning strategy exists.

Proof: If there is a winning control strategy for s, then $s \in \bigcup_i W_i$. Let $s \in W_k$. Let Q_F be an evaluating strategy and Q_G any disturbance strategy.

If $k = 1$ then $Q_F(s)(s) \in W$ indicating Q_F is a winning strategy.

Let the theorem be true for $s \in \bigcup_{i=1}^{k} W_i$. Let $s \in W_{k+1}$. Then by definition of evaluating strategies if $Q_F(s)(s) = s_1$ and $Q_G(s_1)(s_1) = s'$, then $s' \in \bigcup_{i=1}^{k} W_k$. By induction hypothesis there exists sequence $(f_1, \ldots f_n)$ and $(g_1, g_2, \ldots g_{n-1})$ dictated Q_F and Q_G such that $f_n(g_{n-1}(\ldots g_1(f_1(s)\ldots) \in W$. Hence

$$f_n(g_{n-1}(\ldots g_1(f_1(Q_G(s_1)(Q_F(s)(s)) \ldots) \in W.$$

Since Q_G is arbitrary Q_F is a winning strategy for s.

One can prove also an analog of theorem 2.6 regarding evaluating strategies to show that if in every situation one applies a move dictated by some evaluating strategy, the resulting behavior of the game corresponds to that dictated by an evaluating strategy. The theorem and its proof is omitted since these are exact analogs of theorem 2.6 and no new difficulty is created by the relaxed structure of board games.

The idea of evaluations and evaluating strategies are analogs of the similar idea for W-problems. However, certain classes of sets exist for board games whose descriptions also help in the construction of winning strategies and whose analogs do not exist for W-problems. These will now be discussed, for their role in strategy construction as well as for bringing the theory in line with certain graph-theoretic concepts which will be of value in later discussions.

One can define a class of subsets $\{K_i\}$ of $\bigcup_{g \in G} S_g$ as follows: $s \in \bigcup_{g \in G} S_g$ is a member of K_1 if and only if for all g such that $s \in S_g$, $g(s) \in W_1$.

$s \in \bigcup_{g \in G} S_{g_1}$ is a member of K_i $(i > 1)$ if and only if for all g such that $s \in S_g$, $g(s) \in \bigcup_{k=1}^{i} W_k$ and $s \notin \bigcup_{k=1}^{i-1} K_k$.

The following are easy to see.

<u>Lemma 3.11</u> $s \in W_k$, if and only if there exists an $f \in F$ such that $f(s) \in K_{k-1}$ and there is no $f \in F$ such that $f(s) \in K_j (j < k - 1)$.

The proof will be omitted.

Theorem 3.12 Given an $s \in \bigcup_{f \in F} S_f$ if there exists an $f \in F$ such that $f(s) \in K_i$ and no $f' \in F$ such that $f'(s) \in K_j (j < i)$, then $f = Q_F(s)$ for some evaluating strategy Q_F.

The proof of this follows from Lemma 3.11. The importance of $\{K_i\}$ for the construction of winning strategies lie in the fact that if one has descriptions of K_i for every i, then one can construct evaluating strategies also.

Before going on to another very important property of the class $\{K_i\}$ in the next section, it will be useful and worthwhile to indicate an analog of cautious strategies in board games. For this one needs the following definition.

Given a board game, one defines a relation $R \subseteq S \times S$ as follows

sRs' if and only if there exists an $h \in F \cup G$ such that $h(s) = s'$.

A board game is called Progressively Finite if and only if there is no infinite chain $s_1, s_2 \ldots (s_i \in S)$ such that for each i $s_i R s_{i+1}$.

A board control strategy is called cautious if and only if for each $s \in \bigcup_i W_i$ $Q_F(s) = f$ is such that either $f(s) \in W$ or $f(s) \in \bigcup_{g \in G} S_g$ and for all $g \in G$ such that $f(s) \in S_g$, $g(f(s)) \in \bigcup_i W_i$. Evidently every evaluating strategy is a cautious strategy. However, one can say more.

Theorem 3.13 If a board game is progressively finite then a cautious strategy is a winning strategy for every element $s_o \in \bigcup_i W_i$.

Proof: Let Q_F be a cautious control strategy. For any arbitrary control strategy define a sequence $s_o, s_1, s_2 \ldots$ such that for each i

$$s_{i+1} = Q_F(s_i)(s_i) \text{ if } i \text{ is even}$$

$$s_{i+1} = Q_G(s_i)(s_i) \text{ if } i \text{ is odd}$$

Evidently, in this sequence $s_{i+1}Rs_i$ for every i. Since the game is progressively finite, this chain has a last element s_k. Now $s_k \notin \bigcup_{g \in G} S_g$ since otherwise $Q_G(s_k)$ would be defined and s_k would not be the last element of the chain. Similarly $s_k \notin \bigcup_{f \in F} S_f$.

If $s_{k-1} \in \bigcup_{g \in G} S_g$, then $Q_F(s_{k-2})(s_{k-2}) = s_{k-1}$ and hence k-2 is even. However since $s_o \in \bigcup_i W_i$ and S_F is cautious, $s_j \in \bigcup_i W_i$ for all even j. Hence $s_{k-2} \in \bigcup_i W_i$. Hence by definition of cautious strategy $s_k \in \bigcup_i W_i$ which contradicts s_k being the last element of the chain. Hence $s_{k-1} \notin \bigcup_{g \in G} S_g$. Also k-1 is then even. So $s_{k-1} \in \bigcup_i W_i$. Then $s_k \in W$ or $s_k \in \bigcup_{g \in G} S_g$. $s_k \notin \bigcup_{g \in G} S_g$. Hence $s_k \in W$.

Since Q_G is arbitrary, Q_F is a winning strategy.

This theorem is an analog of Theorem 2.8 and indicates that in a finitely progressive game one can construct a winning strategy whenever a description of $\bigcup_i W_i$ is available.

It can also be seen quite easily that a cautious strategy can also be constructed from a knowledge of $\bigcup_i K_i$.

<u>Theorem 3.14</u> If $s \in \bigcup_{f \in F} S_f$ and there exists an $f \in F$ such that $f(s) \in \bigcup_i K_i$, then $f = Q_F(s)$ for some cautious strategy Q_F.

<u>Proof:</u> Let $f(s) \in K_k$. Then, for every $g \in G$ such that $f(s) \in S_g$, $g(f(s)) \in \bigcup_{i=1}^{k-1} W_i \subseteq \bigcup_i W_i$: Hence a strategy Q_F such that $Q_F(s) = f$ is a cautious strategy.

Some of the above theorems could have been strengthened. Also, some further theorems can be added regarding the relationships between W_i and K_i. Also ananlogs of these theorems exists for $\{L_i\}$, the L-evaluations. However, for the present purposes, these are not of immediate importance. In the next section certain well-known graph-theoretic properties of $\bigcup_i K_i$ will be introduced which lead to important methods for construction of winning strategies. In these, attention will be mostly limited to progressively finite board games.

7. Strategies Based on Graph Decomposition

Most of the results in this section are interpretations of well-known results in Graph Theory [23]. These interpretations have been aided by certain elementary concepts of Automata Theory [24]. It is strongly surmised by the author that the extension of the techniques discussed in this section will be of help in developing new methods of problem solution.

The introduction of some graph-theoretic notions are in order. A graph is given by a pair $\langle S,R \rangle$, where S is an abstract set and $R \subseteq S \times S$. Clearly for a board game $\langle S,F,G,W,L \rangle$, if R is defined as in the previous section, then $\langle S,R \rangle$ defines a graph. Given a subset $S' \subseteq S$, $R(S')$ is the set of all elements related by R to elements of S'. In symbols

$$R(S') = \{ s \mid (\exists s')(s' \in S' \text{ and } s'Rs) \quad .$$

$R(\{ s \})$ will be denoted by $R(s)$ for simplicity.

Given a graph $\langle S,R \rangle$, a subset S' S is called a Kernel if

$$s \notin S \text{ implies } R(s) \cap S' \neq \emptyset$$

and

$$R(S') \cap S' = \emptyset$$

Given a graph $\langle S,R \rangle$, an integer valued function $M: S \to N$ mapping S into non-negative integers is called a Grundy function if it has the following property;

$M(s) = n$ implies for all $s' \in R(s)$, $M(s') \neq M(s)$ and for each integer $m < n$ there exists an $s' \in R(s)$ such that $M(s') = m$.

Theorem 3.15 (Berge) If the graph $\langle S,R \rangle$ corresponding to a board game $\langle S,F,G,W,L \rangle$ possesses a Grundy function M, then the set

$$\{ s \mid M(s) = 0 \}$$

is a Kernel.

<u>Proof</u>: If $M(s) = 0$, then for all $s' = h(s) \{h \in F \cup G\}$, $M(s') \neq 0$. Also if $M(s) \neq 0$, then there exists at least some $h \in F \cup G$ such that $M(h(s)) = 0$.

The next theorem like the last one is an obvious specialization of a general theorem in graph theory. One initially introduces another definition:

A progressively finite graph S, R is called <u>progressively bounded</u> if for each $s \in S$ there is an integer $N(s)$ such that all chains of R starting at s has a length less than N.

<u>Theorem 3.16</u> If a board game $\langle S,F,G,W,L \rangle$ is progressively bounded and $\langle S,R \rangle$ is its corresponding graph, then $\langle S,R \rangle$ has an unique Grundy-function M.

<u>Proof:</u> Define a subset G of S as follows:

$$G_o = S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$$

By definition if $s \in G_o$ and M is a Grundy function, $M(s) = 0$. Also G is non-empty by definition.

Since the graph $\langle S,R \rangle$ is progressively bounded for every $s \in S$ there is an integer N such that any chain $s_o, s_1, \ldots s_n$ of members of S such that $s_o = s$ and for each i $s_i \; R \; s_{i+1}$ has length less than or equal to N. The lengths of all chains starting at s is thus bounded above and hence there is a chain of maximal length starting at s. Let the length of the chain of maximal length be $k(s)$. It will be proved by induction that for all integers n if $k(s) = n$ then the value of the Grundy function of s is defined, finite and unique.

If $k(s) = 1$, then there exists a function $h \in F \cup G$ such that $h(s) \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$ and there is no function $h \in F \cup G$ such that $h(s) \in \bigcup_{h \in F \cup G} S_h$, because if there were, there would be a chain starting at s of length greater than 1. Hence $k(s) = 1$ implies $M(s) = 1$.

Let the theorem be true for all s such that $k(s) \leq n$. Let $k(s) = n+1$. Then for any $h \in F \cup G$ such that $s \in S_h$, $k(h(s)) \leq n$ since otherwise there would be a chain

starting at s of length greater than $n+1$. Since the set of values $M(s')$ for all s' with $k(s) \leq n$ is defined and finite, one has an unique integer $M(s)$ such that $M(s) \neq M(s')$ for all s' such that $s' = h(s)$ for some $h \in F \cup G$ and for each $k < M(s)$, there is an s' such that $s' \in h(s)$ for some $h \in F \cup G$ and $M(s') = k$.

The next theorem exhibits the relationship between the class $\{K_i\}$ and the graph theoretic concepts developed earlier.

Theorem 3.17 In a progressively bounded board game $\langle S,F,G,W,L \rangle$ let S' be the set of all points for which the Grundy function M has value 0. Then

$$S' \cap (\bigcup_{g \in G} S_g) \supseteq \bigcup_i K_i.$$

Moreover if $W \cup L = S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$ and if for each $s \in S$, the set $G_s = \{g \mid g \in G \text{ and } s \in S_g\}$ is finite, then

$$S' \cap (\bigcup_{g \in G} S_g) = \bigcup_i K_i$$

Proof: Let $s \in K_k$. It will be shown by induction on k that $M(s) = 0$.

It is clear that if $s \in W_1$, then $M(s) > 0$, since there is an $f \in F$ such that $f(s) \in W$ and hence $M(f(s)) = 0$. Now if $s \in K_1$, then $R(s) \subseteq W_1$. Hence for all g such that $s \in S_g$, $M(g(s)) > 0$. Hence $M(s) = 0$.

Let the theorem be true for $s \in K_i$. If $s \in W_{i+1}$, then $M(s) > 0$ since there is an $f \in F$ such that $F(s) \in K_i$ (by Lemma 3.11) and hence $M(f(s)) = 0$. Let $s \in K_{i+1}$, then by definition, for every $g \in G$ such that $s \in S_g$, $g(s) \in W_{i+1}$, whence $M(s) = 0$.

Hence $\bigcup_i K_i \subseteq S'$. Also by definition $\bigcup_i K_i \subseteq \bigcup_{g \in G} S_g$. Hence the first part of the theorem follows.

For the second part of the theorem, let $s \in S' \cap (\bigcup_{g \in G} S_g)$. Define any control strategy Q_F such that for all $s \in \bigcup_{f \in F} S_f$, if $M(s) \neq 0$, $Q_F(s)(s) \in S'$. Since S' is a Kernel, such a strategy exists. Let Q_G be any disturbance strategy. Let $Q_G(s)(s) = s'$. Define the sequence $h_1, h_2, \ldots$ of members of F and G as follows

$$h_1 = Q_F(s'); \; h_2 = Q_G(h_1(s'))$$

and in general

$$h_{2i+1} = Q_F(h_{2i}(h_{2i-1} \dots h_1(s') \dots)$$

$$h_{2i} \;\; = Q_G(h_{2i-1} \dots h_1(s') \dots).$$

Since the game is progressively finite, such a sequence must end. Let h_m be the last element of the sequence. Then $h_m \notin G$. If it were, then $M(h_m(h_{m-1} \dots h_1(s') \dots) = 0$ and $h_{m-1} \in F$. But by the definition of h_{m-1} and the property of Q_F, $M(h_{m-1}(\dots h_1(s')) = 0$. This leads to a contradiction. Hence $h_m \in F$ and $h_m(h_{m-1} \dots (h_1(s') \dots) \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g = W \cup L$ and by B4 $h_m(h_{m-1} \dots (h_1(s') \dots) \notin L$. Hence $h_m(h_{m-1} \dots (h_1(s') \dots) \in W$. Since Q_G is arbitrary, Q_G is a winning strategy for s'. Hence $s' \in \bigcup_i W_i$ by corollary 3.9. Let $s' \in W_j$. Since $s' = g(s)$ for an arbitrary g such that $s \in S_g$, one obtains that if $s \in S' \cap (\bigcup_{g \in G} S_g)$, then for an $g \in G$ such that $s \in S_g$, $g(s) \in W_j$ for some j.

Given $s \in S' \cap (\bigcup_{g \in G} S_g)$ define the set of integers

$$N_s = \{ j \mid (\exists g)(s \in S_g \text{ and } g(s) \in W_j) \} \;.$$

Since by assumption there is only a finite set of disturbances $g \in G$ such that $s \in S_g$, N_s is a finite set. Let k be the maximum of N_s. Then for all g such that $s \in S_g$, $g(s) \in \bigcup_{i=1}^{k} W_i$. Hence $s \in \bigcup_{i=1}^{k} K_i$. This proves

$$S' \cap (\bigcup_{g \in G} S_g) \subseteq \bigcup_i K_i$$

and with the reverse inequality proven in the first part of the theorem it proves the second part of the theorem.

A board game will be called <u>Grundy-Tractable</u> if and only if it is progressively finite, if $W \cup L = S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$ and for any $s \in \bigcup_{g \in G} S_g$ the set $G_s = \{ g \mid g \in G \text{ and } s \in S_g \}$ is finite. The calculation of the Grundy-Function in a

Grundy Tractable board game leads to a mode of description for $\bigcup_i K_i$ and hence to the construction of winning strategies. However, the method for calculating the Grundy-Function as indicated in theorem 3.16 is certainly not a very practicable method. Practicable methods are available for a certain class of board games which will now be discussed.

A board game $\langle S,F,G,W,L\rangle$ will be called Graph-Interpretable if and only if there exists an abstract set Ω , a set H of functions mapping subsets of Ω into Ω (i.e. $h \in H$ implies $h:S_h \to \Omega$ where $S_h \subseteq \Omega$) and a subset T of Ω such that

Gr1) $S = \Omega \times \{0,1\}$

Gr2) i) $W = \{(s,1) \mid s \in T\}$

ii) $L = \{(s,0) \mid s \in T\}$

Gr3) $h \in H$ if and only if i) there exists an $f \in F$ such that $s \in S_h$ if and only if $(s,0) \in S_f$ and $f(s,0) = (h(s),1)$

and ii) there exists a $g \in G$ such that $s \in S_h$ if and only if $(s,1) \in S_g$ and $g(s,1) = (h(s),0)$

Gr4) i) $(s,k) \in S_f$ and $f \in F$ implies $k = 0$

ii) $(s,k) \in S_g$ and $g \in G$ implies $k = 1$.

The rest of the discussion in this section will be restricted to graph-interpretable games--to Grundy Tractable graph-interpretable games in particular. Obviously, a graph-interpretable game is completely specified by the triple $\langle \Omega, H, T\rangle$ and defines a graph $\langle \Omega, \bigcup_{h \in H} h\rangle$. The Grundy-Function M of this graph has the property that $s \in T$ implies $M(s) = 0$.

Given a finite set of Graph-Interpretable games $\{\langle \Omega_i, H_i, T_i\rangle \mid 1 \leq i \leq n\}$, a graph-interpretable game $\langle \Omega, H, T\rangle$ is called the sum of $\{\langle \Omega_i, H_i, T_i\rangle\}$ if and only if

S1) $\Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$

S2) $h \in H$ and $h(s_1,\ldots s_n) = (s_1',\ldots,s_n')$ if and only if there is an unique positive integer $i \leq n$ and member $h_i \in H_i$ such that

$s_j' = s_j$ if $j \neq i$

$s_i' = h_i(s_i)$

S3) $T = T_1 x T_2 x \ldots x T_n$

Theorem 3.18 Given a finite set $\{\langle\Omega_i, H_i, T_i\rangle\}$ of Grundy-Tractable Graph-Interpretable Games, their sum is Grundy-Tractable.

Proof: For each i, $W_i \cup L_i = S_i - \bigcup_{f \in F_i} S_f - {}_{g\ G_i} S_g$, since the game $\langle\Omega_i, H_i, T_i\rangle$ is Grundy-Tractable. If now $s \in \Omega_i - \bigcup_{h \in H_i} S_h$ then there is no $f \in F_i$ such that $(s,0) \notin S_f$ and by definition no $g \in G_i$ such that $(s,0) \in S_g$. Hence $(s,0) \in S_i - \bigcup_{f \in F_i} S_f - \bigcup_{g \in G_i} S_g$. Hence $(s,0) \in L_i$ or $s \in T_i$. Hence $T_i \supseteq \Omega_i - \bigcup_{h \in H_i} S_h$. That $T_i \subseteq \Omega_i - \bigcup_{h \in H_i} S_h$ follows from (B3). Hence for each i, $T_i = \Omega_i - \bigcup_{h \in H_i} S_h$. Let $\langle\Omega, H, T\rangle$ be the sum of $\langle\Omega_i, H_i, T_i\rangle$ $(i = 1,2,\ldots n)$. If $(s_1, s_2, \ldots s_n) \in T$, then for each i, $s_i \in \Omega_i - \bigcup_{h \in H_i} S_h$. Hence there is no $h \in H$ such that $s \in S_h$. Hence $T \subseteq \Omega - \bigcup_{h \in H} S_h$. Similarly if $(s_1, s_2, \ldots s_n) \in \Omega - \bigcup_{h \in H} S_h$, then for each i $s_i \in \Omega_i - \bigcup_{h \in H_i} S_h$. Since $T_i = \Omega_i - \bigcup_{h \in H_i} S_h$, $s_i \in T_i$. Hence $(s_1, \ldots s_n) \in T_1 x T_2 x \ldots x T_n = T$. Hence $\Omega - \bigcup_{h \in H} S_h \subseteq T$. This with the previous inequality shows $T = \Omega - \bigcup_{h \in H} S_h$. Hence, $W \cup L = (\bigcup_{s \in T}\{(s,0)\}) \cup (\bigcup_{s \in T}\{(s,1)\}) = S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$.

Let $s \in \bigcup_{g \in G} S_g$. Hence $s = (s,1)$ for some $s = (s_1, s_2, \ldots s_n) \in \Omega$ and for some $i (1 \leq i \leq n)$, $s_i \in \bigcup_{h \in H_i} S_{h_i}$. Since the game $\langle\Omega_i, H_i, T_i\rangle$ is Grundy-Tractable, the set $\{h \mid h \in H_i$ and $s \in S_h\}$ is finite. Hence the set of h such that $(s_1, s_2, \ldots s_n) \in S_h$ is finite. Hence the set of $g \in G$ such that $(s,1) \in S_g$ is finite.

To show that $\langle \Omega, H, T \rangle$ is progressively finite, one assumes to the contrary that there exists an infinite sequence $h^1, h^2 \ldots$ of functions in H such that for and $s \in \Omega$, $s \in S_{h^1}$, and in general $h^{i-1}(h^{i-2}(\ldots h^1(s)\ldots) \in S_{h^i}$.

One says that $\langle \Omega_k, H_k, T_k \rangle$ occurs in h^i if the k^{th} component of $h^{i-1}(\ldots h^1(s)\ldots)$ and the k^{th} component of $h^i(h^{i-1}(\ldots h^1(s)\ldots)$ are distinct. Since one game $\langle \Omega_k, H_k, T_k \rangle$ occurs in h^i for every i and there are only n games, some game $\langle \Omega_k, H_k, T_k \rangle$ must occur in h^i for an infinite subsequence $h^{i_1}, h^{i_2}, \ldots$ of $\{ h^i \}$. Then the k^{th} component s' of $h^{i_1-1}(h^{i_1-2} \ldots h^1(s)\ldots)$ belongs to $S_{h_{i_1}}$ for some $h_{i_1} \in H_k$ and the k^{th} component of $h^{i_1}(h^{i_1-1}(\ldots h^1(s)\ldots)$ is $h_{i_1}(s')$ and is identical to the k^{th} component of $h^{i_2-1}(h^{i_2-2}(\ldots h'(s)\ldots)$ and is a member of $S_{h_{i_2}}$ for some $h_{i_2} \in H_k$. One thus obtains an infinite sequence $h_{i_1}, h_{i_2} \ldots$ of members of H_k such that some element s' of Ω_k belongs to $S_{h_{i_1}}$, and for each p, $h_{i_p-1}(h_{i_p-2} \ldots h_{i_1}(s')\ldots) \in S_{h_{i_p}}$. This contradicts the assumption that $\langle \Omega_k, H_k, T_k \rangle$ is Grundy-Tractable and hence progressively finite.

The importance of the sum of Graph-Interpretable games stems from the fact that if the Grundy-Functions of the components are known, then the Grundy-Function of the sum can be calculated quite readily.

To indicate the method of this calculation one needs to define a special binary operation $\oplus$ between non-negative integers. Let a and b be two such integers. Let

$$a = a_0 + a_1 2 + a_i 2^2 + \ldots a_m 2^m \quad 0 \leq a_i \leq 1$$

$$b = b_0 + b_1 2 + b_i 2^2 + \ldots b_n 2^n \quad 0 \leq b_i \leq 1$$

i.e. let $a_m a_{m-1} \ldots a_0$ and $b_n b_{n-1} \ldots b_0$ be the binary representation a and b. One can assume without loss of generality that $m = n$ and that some of the leading binary digits are 0.

One defines

$$c = (a \oplus b) = c_0 + c_1 2 + c_2 2^2 + \ldots \; c_m 2^m$$

where for each i, $c_i = a_i + b_i$ (mod. 2).

It can be seen easily that the $\oplus$ operation is a group operation on integers, with 0 as the unit element and every integer its own inverse.

The following theorem indicates the use of the $\oplus$ operator in the calculation of the Grundy-Functions of sums of games.

Theorem 3.19 Let $\langle \Omega_i, H_i, \tau_i \rangle$ $i = 1,2,\ldots n$ be a collection of Graph-Interpretable. Grundy-Tractable games and let M_i be their Grundy functions. One defines M on their sum as follows

$$M((s_1,\ldots s_n)) = M_1(s_1) \oplus M_2(s_2) \oplus \ldots \oplus M_n(s_n).$$

M is a Grundy-Function on the sum.

Proof: Let $M((s_1,s_2,\ldots s_n)) = k$ and let $M_i(i_i) = n_i$. One has to show that for all $h \in H$ such that $(s_1 \ldots s_n) \in S_h$, $M(h((s_1,s_2,\ldots s_n))) \neq k$ and for each integer $m < k$ there exists an $h \in H$ such that $(s_1,\ldots s_n) \in S_h$ and $M(h((s_1,\ldots s_n))) = m$.

Let $k = k_0 + k_1 2 + \ldots \; k_t 2^t$

and for each i $n_i = n_{i0} + n_{i1} 2 + \ldots n_{it} 2^t$.

Let $h((s_1,s_2,\ldots s_n)) = (s_1', s_2',\ldots s_n')$.

Then there exists a j such that $s_i = s_i'$ for all $i \neq j$ and $s_j' = h_j(s_j)$ for some $h_j \in H_j$. Then $M_j(s_j') \neq n_j$. Now $M((s_1', s_2'\ldots s_n')) = n_1 \oplus n_2 \oplus \ldots \oplus n_{j-1} \oplus M_j(s_j') \oplus n_j \oplus \ldots n_n$ and $M((s_1,s_2\ldots s_n)) = k = n_1 \oplus n_2 \oplus \ldots \oplus n_j \oplus \ldots n_n$, whence

$$M((s_1,s_2,\ldots s_n)) \; + \; M \;((s_1',s_2',\ldots s_n')) = n_j \oplus M_j(s_j').$$

Since $M_h(s_j') \neq n_j$, $M((s_1,s_2,\ldots s_n)) \neq M((s_1',s_2',\ldots s_n'))$, proving that for all h such that $(s_1,s_2,s_n) \in S_h$, $M(h((s_1,s_2,\ldots s_n))) \neq M((s_1,s_2,\ldots s_n)$.

Let $m < k$.

Let $m = m_0 + m_1 2 + \ldots + m_t 2^t$. There must be at least one j such that $m_j \neq k_j$

since $m \neq k$. Let j_0 be the largest such integer. Since $m < k$, $m_{j_0} = 0$ and $k_{j_0} = 1$. Since $k_{j_0} = 1$, there is an i_0, such that $n_{i_0 j_0} = 1$. Define an integer $n'_{i_0} = n'_{i_0 0} + n'_{i_0 1}\, 2 + \ldots n'_{i_0 t} 2^t$ as follows:

$$n'_{i_0 j} = n_{i_0 j} \text{ if } m_j = k_j$$

$$n'_{i_0 j} = n_{i_0 j} + 1 (\text{mod. } 2) \text{ if } m_j \neq k_j.$$

Since $m_j = k_j$ for $j > j_0$ and $m_{j_0} \neq k_{j_0}$ and $n_{i_0 j_0} = 1$, we have

$$n'_{i_0 j} = n_{i_0 j} \text{ for } j > j_0.$$

Also

$$n'_{i_0 j_0} < n_{i_0 j_0} \text{ since } n'_{i_0 j_0} = 0.$$

Hence $n'_{i_0} < n_{i_0}$. Moreover, for each $j < t$,

$$k_j = n_{1j} + n_{2j} + \ldots n_{(i_0-1)j} + n_{i_0 j} + n_{(i_0+1)j} + \ldots n_{nj} \text{ (mod.2)}.$$

Consider

$$p_j = n_{1j} + n_{2j} + \quad + n_{(i_0-1)j} + n'_{i_0 j} + n_{n_j} \text{ (mod. 2)}$$

whence $p_j + k_j = n_{i_0 j} + n'_{i_0 j}$

whence $p_j = k_j$ if and only if $n_{i_o j} = n'_{2_o k}$ i.e. if and only if $m_j = k_j$. Hence $p_j = m_j$ for every j. Hence

$$m = n_1 \oplus n_2 \oplus \ldots \oplus n_{(i_0-1)} \oplus n'_{i_0} \oplus (n_{(i_0+1)} \cdot \oplus n_n.$$

Since $n'_{i_0} \quad n_{i_0} = M_{i_0}(s_{i_0})$, there exists an $s'_{i_0} \in \Omega_{i_0}$ and an $h \in H_{i_0}$ such that $M_{i_0}(s'_{i_0}) = n'_{i_0}$ and $h(s_{i_0}) = s'_{i_0}$. Hence there exists an $h' \in H$ such that $h((s_1, s_2, \ldots s_n)) = (s_1, s_2, \ldots s_{i_0-1}, s'_{i_0}, s_{i_0+1}, \ldots s_n)$ and $M((s_1, s_2, \ldots s_{i_0-1}, s'_{i_0}, s_{i_0+1}, \ldots s_n)) = m$.

If one is given a graph-interpretable game $\langle \Omega, H, T \rangle$ which is the sum of a finite number of graph-interpretable, Grundy-Tractable Games, then one can calculate the Grundy-function of $\langle \Omega, H, T \rangle$ identify the set $\bigcup_i K_i$ for it and construct a winning strategy. This necessitates a knowledge of the Grundy functions of

each component. However, since the component games have much fewer states, their Grundy-functions may be calculated by the exhaustive technique indicated in Theorem 3.16.

This technique of construction for winning strategies, of course, is limited to graph-interpretable games which can be decomposed into Grundy-tractable games. In what follows conditions will be set down for the graph-interpretability of board games and decomposability of graph-interpretable games.

Theorem 3.20 A board game $\langle S,F,G,W,L\rangle$ is graph interpretable if and only if there are two subsets S_0, S_1 of S and two one-one maps

$$\alpha : S_0 \to S_1$$

$$\beta : F \to G$$

such that

i) $L \subseteq S_0$; $W \subseteq S_1$

ii) $\bigcup_{g \in G} S_g \subseteq S_1$; $\bigcup_{f \in F} S_f \subseteq S_0$

iii) $\alpha^{-1}(S_g) = S_{\beta^{-1}(g)}$; $\alpha(S_f) = S_{\beta(f)}$

iv) $\alpha^{-1}(f(s)) = \beta(f)(\alpha(s))$; $\alpha(g(s)) = \beta^{-1}(g)(\alpha^{-1}(s))$

for each $f \in F$, $g \in G$ and any $s \in S$ for which either side of the equation is defined.

v) $\alpha(L) = W$

vi) $S_0 \cup S_1 = S$

Proof: Let $\langle S,F,G,W,L\rangle$ be Graph-Interpretable. There exists a set Ω a set H of partial functions mapping subsets of Ω into Ω and a subset T of Ω satisfying Gr1 to Gr4. Define $S_0 = \{(s,0) \mid s \in \Omega\}$ and $S_1 = \{(s,1) \mid s \in \Omega\}$. Then (vi) is satisfied.

Define $\alpha(s,0) = (s,1)$ for each element of S_0. This is one-one from S_0 onto S_1. Also $(s,k) \in S_g$ for $g \in G$ implies $k = 1$ whence $(s,k) \in S_1$. Hence $\bigcup_{g \in G} S_g \subseteq S_1$. Similarly $\bigcup_{f \in F} S_f \subseteq S_0$ satisfying (ii).

$$W = \{(s,1) \mid s \in T\} \subseteq \{(s,1) \mid s \in \Omega\} = S_1.$$

Similarly $L \subseteq S_0$ proving (i).

If $(s,0) \in L$ then $s \in T$ and $\alpha(s,0) = (s,1)$ whence $\alpha(s,0) \in \{(s,1) \mid s \in T\} = W$. Hence $\alpha(L) \subseteq W$. It can be shown similarly that $\alpha^{-1}(W) \subseteq L$ proving (v).

For each $f \in F$ there exists an $h \in H$ and $g \in G$ such that $S_f = \{(s,0) \mid s \in S_h\}$ and $S_g = \{(s,1) \mid s \in S_h\}$, $f(s,0) = (h(s),1)$ and $g(s,1) = (h(s),0)$. If one defines $\beta(f) = g$, the resulting β is one-one onto. Also (iii) and (iv) will be satisfied by this β.

Let now $\langle S,F,G,W,L\rangle$, S_0, S_1, α and β be as defined by (i) through (vi). Then one can define Ω, H and T as follows: $\Omega \subseteq S \times S$ such that $(s_0,s_1) \in \Omega$ if and only if $s_1 = \alpha(s_0)$. Hence $s_0 \in S_0$ and $s_1 \in S_1$. Denote s_0 by $\{(s_0,s_1), 0\}$ and s_1 by $\{(s_0,s_1),1\}$. Since α is one-one onto, and because of (vi), if $s \in S$ then either $s \in S_1$ and $s = \{(\alpha^{-1}(s),s), 1\}$ or $s \in S_0$ and $s = \{(s,\alpha(s)), 0\}$. Hence $S = \Omega \times \{0,1\}$ satisfying Gr1.

$s \in L \subseteq S_0$ if and only if $\alpha(s) \in W \subseteq S_1$. Define by T the set of all pairs $(s,\alpha(s))$ such that $s \in L$. Then $s \in T$ if and only if $(s,0) \in L$ and $(s,1) \in W$. This establishes Gr2.

$H \subseteq \{h \mid h: \Omega_h \to \Omega\}$ is constructed as follows. Let $f \in F$ and $s \in S_f$. Then by (ii) and the construction of Ω, $s = \{(s,\alpha(s)),0\}$. By (iii) $\alpha(s) \in S_{\beta(f)}$. Define h so that

$$S_h = \{(s,\alpha(s)) \mid s \in S_f\} = \{(s,\alpha(s)) \mid \alpha(s) \in S_g\}$$

and $h(s,\alpha(s)) = (\beta(f)(\alpha(s)), f(s))$; $h(s,\alpha(s)) \in \Omega$ since by (iv) above $\alpha(\beta(f)(\alpha(s))) = f(s)$. Also, $f(s) = (h(s,\alpha(s)),1)$. This establishes first part of Gr3 and Gr4. The second parts follow similarly.

This theorem is included to show clearly what kind of symmetry is demanded of a graph-representable game. The definition, based on the existence of the Ω graph, did not clarify the structure sufficiently.

It may be worthwhile pointing out at this point that the Nim-type games are graph-interpretable. This follows from the definition of graph interpretable games, the role of Ω being played by the set of sequences I. The reader can verify that conditions G1-G4 are satisfied.

The Tic-tac-toe-like games are graph-interpretable also if there is a premutation P on N such that $A \in \mathcal{A}$ if and only if there exists a $B \in \mathcal{B}$ such that $n \in A$ if and only if $P(n) \in B$. In this case one can set $S_0 = \{(s,p) \mid p = 0\}$ and $S_1 = \{(s,p) \mid p = 1\}$. The function α may be defined as follows: $((s,0)) = (s',1)$ where $s'(P(n)) = X$ if $s(n) = Y$, $s'(P(n)) = Y$ if $s(n) = X$ and $s'(n) =$ otherwise; is defined by $(n,X) = (P(n),Y)$. At this point it may not be worthwhile proving formally that the partition S_0, S_1 and the maps and satisfy the conditions i - vi in theorem 3.20. However, the reader will do well to convince himself, at least intuitively that this is so.

It can be strongly surmised that games like chess and checkers are also graph-representable in this sense.

Theorem 3.21 A graph-representable game $\langle \Omega, H, T \rangle$ is the sum of a set of n graph-representable games $\{\langle \Omega_i, H_i, T_i \rangle \mid 1 \leq i \leq n\}$ if and only if there exists a set of n equivalence relations $\{E_i \mid 1 \leq i \leq n\}$ on Ω and a set of disjoint subsets $\{H_i' \mid 1 \leq i \leq n\}$ of H such that

i) $\bigcup H_i' = H$

ii) $\bigcap E_i = I$, the identity relation on Ω

iii) For any $s_1 s_2 \ldots s_n$ $(s_i \in \Omega)$ there exists $s \in \Omega$ such that $s_i E_i s$ for each i.

iv) $s \notin T$ implies that for some E_i $\quad sE_is'$ implies $s' \notin T$

v) sE_is' implies $s \in S_h$ if and only if $s' \in S_h$ for all $h \notin H_i'$ and $h(s)E_ih(s')$ for all h such that $s \in S_h$ and $s' \in S_h$.

vi) $h \in H_i'$ implies for all $s \in S_h$, $h(s)E_js$ for all $j \neq i$.

Proof: The sufficiency is proved by constructing $\{\langle\Omega_i, H_i, T_i\rangle\}$ as follows:

Ω_i is a set of equivalence classes of E_i. Since $\bigcap E_i = I$, two distinct elements of Ω does not lie in the same equivalence class of every E_i. Hence $\Omega \subseteq \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$. Let $s \in \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$. If $s \notin \Omega$, then there is some set of equivalence classes $e_1, e_2, \ldots e_n$ (e_i an equivalence class of E_i for each i) such that $e_1 \cap e_2 \ \ldots \cap e_n = \emptyset$. Take an element $s_1 \in e_1$, $s_2 \in e_2, \ldots s_n \in e_n$. But there exists s(by(iii) above) such that $s \in e_1 \cap e_2 \ \ldots \cap e_n$. This contradicts $e_1 \cap e_2 \cap \ldots \cap e_n = \emptyset$. Hence $s \in \Omega$, proves $\Omega_1 \times \Omega_2 \times \ldots \times \Omega_n \subseteq \Omega$. S1 is thus established.

Let T_i be the set of equivalence of E_i which contain some element $s \in T$. Clearly, then $T \subseteq T_1 x T_2 x \ldots x T_n$, again since $\bigcap E_i = I$. Let $s \notin T$; for each i let e_i be the equivalence class of E_i continaing s. By (iv), there exists some E_i such that $s' \notin T$ for no member s' of e_i. Hence $e_i \notin T_i$. Hence $s \notin T_1 x T_2 x \ldots x T_n$. Hence the complement of T is contained in the complement of $T_1 x T_2 x \ldots x T_n$ or $T_1 x T_2 x \ldots T_n \subseteq T$. This establishes S3.

Define H_i as follows. For every $h' \in H_i'$, define a member $h \in H_i$, such that if $s \in S_{h'}$, then the equivalence class e_i of E_i containing s is a member of S_h and $h(e_i)$ is the equivalence class of E_i containing $h'(s)$. By (v), this determines the function h inequivocally. For all E_j $(j \neq i)$, if e_j is the equivalence class of E_j containing s, then by (vi) e_j is also the equivalence class containing $h'(s)$. Hence if $h' \in H_i'$, then the $h'(s)$ is in the intersection of the blocks of $E_j(j \neq i)$ containing s and the block $h(e_i)$. This establishes S2 indicating that conditions (i)-(vi) is sufficient for (S1)-(S3).

To show necessity let $\langle\Omega, H, T\rangle$ be the sum of $\{\Omega_i, H_i, T_i\}$ $(1 \leq i \leq n)$. Define $(s_1, s_2, \ldots s_n)\ E_i (s_1', s_2', \ldots s_n')$ if and only if $s_i = s_i'$. Clearly

$(s_1,s_2,\ldots s_n)(\bigcap E_i)(s_1',s_2',\ldots s_n')$ if and only if $s_i = s_i'$ for every i. This establishes (ii).

Since $T = T_1 \times T_2 \times \ldots \times T_n$, by (S3) $(s_1,s_2,\ldots s_n) \notin T$ implies $s_i \notin T_i$ for some i. Hence for any $(s_1',s_2',\ldots s_n')$ $s_i = s_i'$ implies $(s_1', s_2',\ldots s_n') \notin T$. Hence (iv) follows.

Now for every $h \in H$ there is an unique integer $i(1 \leq i \leq n)$ such that $h(s_1,\ldots s_n) = (s_1',s_2',\ldots s_n')$ implies $s_i \in S_{h'}$ for some $h' \in H_i$, $s_i' = h'(s_i)$ and $s_j' = s$ for all $j \neq i$. Define the class of subsets H_i' as follows: $h \in H_i'$ if and only if the corresponding $h' \in H_i$. Since there is an unique i with this property for every element of H, the subsets H_i' are disjoint. Since an i exists for every element $h \in H$, (i) follows. Also, if $(s_1,s_2,\ldots s_n)$ E_i $(s_1',s_2',\ldots s_n')$, then $s_i = s_i'$. Hence if $(s_1,s_2,\ldots s_n) \in S_h$ for $h \in H_i'$ then $s_i \in S_{h'}$ for $h' \in H_i$ hence $(s_1',s_2',\ldots s_n') \in S_h$ also. Again $h((s_1,s_2,\ldots s_n)) = (s_1,s_2,\ldots s_{i-1}, h(s_i),s_{i+1},\ldots s_n)$ and $h((s_1',\ldots s_n')) = (s_1',s'_2,\ldots s_{i-1}',h(s_i),s_{i+1}',\ldots s_n')$ whence $h((s_1,s_2,\ldots s_n))E_i h(s_1',s_1',\ldots s_n')$. Also, $h((s_i,s_2,\ldots s_n)) = (s_1,s_2,\ldots h(s_i)\ldots s_n)$ so that $h((s_1,s_2,\ldots s_n))E_j$ $(s_1,s_2,\ldots s_n)$ for all $j \neq i$. This establishes (v) and (vi).

Let there be n elements $s_1,s_2,\ldots s_n$ in Ω. Denote these by $(s_{11},\ldots s_{1n})$, $(s_{21},\ldots s_{2n})\ldots(s_{n1}\ldots s_{nn})$ respectively. Let $s' = (s_{11},s_{22},\ldots s_{nn})$. Then $s_i E_i s_i'$ for each i. This establishes (iii).

It must be emphasized at this point that for any application of theorems 3.20 or 3.21 to be practicable, one needs to have descriptions of the blocks of the partitions referred to in these theorems. This again necessitates the use of a language in which such descriptions can be expressed by tractably short expressions. It will be indicated in the next section how some of the Nim-type

games shown in Section 4 are sum-decomposable. In these cases, the descriptions of the equivalence classes of E_i are particularly simple.

In what follows, some of the ideas developed in this and the preceding sections will be exemplified for Nim class of games. Discussions of the Tic-tac-toe class of games will be reserved for a later chapter.

8. Some Examples of Strategy-Construction

Concentrating attention on the Nim-like games, one can quite easily construct a winning strategy for the first game in the examples for all states $(i,0)$ where $i \neq 0 \pmod{k+1}$. It can be seen that $\bigcup_i K_i = ((k+1)p,1)$ for any integer $p > 0$. This is because for all $(x_1,1) \in G$ such that $x_1 \leq k$, $(x_1,1)((k+1)p,1) = ((k+1)p-x_1,0)$. If one chooses $(k+1-x_1,0) \in F$, one obtains $(k+1-x_1,0)((k+1)p-x_1,0) = ((k+1)(p-1),1)$. If $p = 1$, the resulting situation is a member of W so that $(x_1,1)((k+1)\ ,1) \in W_1$ for all $(x_1,1)$ satisfying condition α. The result follows by induction on p. Hence in any situation $(i,0)$ if $i \not\equiv 0 \pmod{k+1}$, one can choose an integer $x \equiv i \pmod{k+1}$ such that $0 < x \leq k$ and such that $(x,0)(i,0) \in \bigcup_i K_i$. Since the game is obviously Progressively Bounded, this yields a winning control strategy for the set of situations mentioned.

For future discussions, it may be worthwhile pointing out that the Grundy-function M of the graph of this game is definable as the smallest integer $M(i)$ such that $M(i) \equiv i \pmod{k+1}$. Figure 3.5 indicates this fact for a game with $i \leq 7$ for all nodes and $k = 2$. The numeral at each node indicates the value of i and the numeral in parenthesis indicates the value of the Grundy-Function

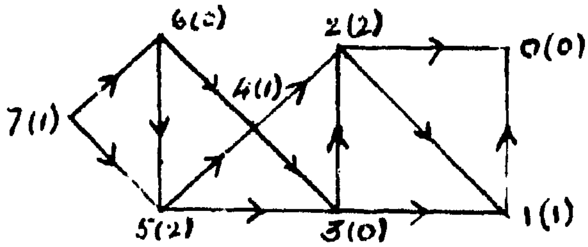


Fig. 3.5

The game with $n = 2$ cited in Section 4, is also analyzable in terms of the Grundy function. However, this does not shed any further light on the contents of this book. It will be analyzed in an entirely *ad hoc* manner.

It can be seen in this case that the situations $((2p,2q),1)$ belong to $\bigcup_i K_i \cup W$ for all p,q. For $q = p = 0$, the situation belongs to W. Also, for all p and q the only situation to which the disturbance can move are $((2p-1,2q),0)$, $((2p,2q-1),0)$ and $((2p-1,2q-1),0)$ from which the control can move to $(2(p-1),2(q-1),0)$. For $\cdot = q = 0$, and $p = 0$, $q = 1$ and $p = 1$, $q = 0$, then $((2p,2q),1)$ is a member of K_1. The result follows by induction.

One can express the above results by saying that if the control can reduce the situation to the case where both heaps are even, then the disturbance has to reduce at least one heap to an odd number from which the control can move always to a "both even" situation.

In both of the above cases the descriptions of $\bigcup_i K_i$ was expressible in a language containing predicates involving ideals of integers modulo fixed integers. However, there was very little indication of an uniform procedure for generating the description. One may say that if one has an efficient "pattern recognition" procedure and a pre-defined knowledge of patterns such as equivalences mod. k, one can recognize these patterns through case studies, generating a theorem (like $((2p,2q),1) \in \bigcup_i K_i$) from the recognized patterns and proving them.

In some fortunate cases, the structures of sets like K_i and W_i become quite transparent; in others, techniques indicated by Theorem 3.20 and 3.21 become effective. This latter can be exemplified by the two last classes of games mentioned in Sec. 3.

One can see that both of these games (n is any finite number, and $\alpha \equiv \beta \equiv (\exists x_i)(j \neq i \rightarrow x_j = 0$ and $x_i > 0)$ in the first case and $\alpha \equiv \beta \equiv (\exists x_i)(j \neq i \rightarrow x_j = 0$ and $k \geq x_i > 0)$ for some specified k in the second case) can be described as sum-compositions of n games. In the first case $(x_i > 0)$ the value Grundy-function of each component graph at each node equals the number of sticks in the heap (this fact can be gleaned from a very simple "pattern recognition", at present

non-mechanizable). In the second case, the value of the Grundy-Function at each node can be calculated by the same method as indicated for the case $n = 1$. Once these Grundy Functions are known, the Grundy functions of the sum game is calculated by the method indicated in theorem 3.19.

A specific example will make the procedure clear. Let us take the case where there are 4 heaps of sticks and each move consists of removing not more than 3 sticks from one of the heaps. Then the value of the Grundy function for the situation $((i_1,i_2,i_3,i_4),p)$ is $M(i_1) \oplus M(i_2) \oplus M(i_3) \oplus M(i_4)$ where $M(i_k)$ is the remainder obtained by dividing i_k by 4. A winning strategy exists for all situations where a control is applicable and the value of the Grundy Function is not zero. For example, for the situation ((7,7,6,5),0) the value of the Grundy-function is $3 \oplus 3 \oplus 2 \oplus 1 = 3$. The value can be reduced to zero, by removing 3 sticks from either the first or the second pile, reducing the situation in the first case to ((4,7,6,5),1). Any disturbance renders the value of the Grundy-function to non-zero. As an example, the disturbance ((0,0,0,2),1) reduces the situation to ((4,7,6,3),0) whose Gruddy function is $0 \oplus 3 \oplus 2 \oplus 3 = 2$. The move ((0,0,0,2),0) reduces the situation to ((4,7,6,1),1) whose Grundy function is $0 \oplus 3 \oplus 2 \oplus 1 = 0$. A typical continuation to the end is shown in Fig. 3.6.

The result of Theorem 3.19 is the strongest one known to the author regarding the calculation of Kernels of game graphs. Other results pertinent to calculation of Grundy-functions of graphs are known; however the calculations are still prohibitively lengthy except in special cases. Results for parallel decomposition of graphs are available only for cases where the structures of the component graphs obey severe restrictions.

Many games do not have evident decompositions of the type exemplified above. However, it is believed that theorem 3.21 and various weaker forms may enable the recognition of decomposability in games which are not evidently decomposable.

$$
\begin{tikzcd}
((7,7,6,5),0) \arrow[d, "{((3,0,0,0),0)}"] & \\
((4,7,6,5),1) \arrow[r, "{((0,0,0,2),1)}"] & ((4,7,6,3),0) \arrow[d, "{((0,0,0,2),0)}"] \\
((1,7,6,1),0) \arrow[d, "{((1,0,0,0),1)}"] & ((4,7,6,1),1) \arrow[l, "{((3,0,0,0),1)}"'] \\
((0,7,6,1),1) \arrow[r, "{((0,0,2,0),1)}"] & ((0,7,4,1),0) \arrow[d, "{((0,2,4,1),0)}"] \\
((0,2,4,1),0) \arrow[d, "{((0,1,0,0),0)}"] & ((0,5,4,1),1) \arrow[l, "{((0,3,0,0),1)}"'] \\
((0,1,4,1),1) \arrow[r, "{((0,0,2,0),1)}"] & ((0,1,2,1),0) \arrow[d, "{((0,0,2,0),0)}"] \\
((0,1,0,0),0) \arrow[d, "{((0,1,0,0),0)}"] & ((0,1,0,1),1) \arrow[l, "{((0,0,0,1),1)}"'] \\
((0,0,0,0),1) &
\end{tikzcd}
$$

Fig. 3.6

One may look for relaxation of these conditions by realizing that the relation R corresponding to the game may be partitioned into various classes of functions and one need not restrict oneself to an unique set H of functions. One can try various partitions of R (as long as the elements of these partitions are easily describable) so that the conditions of Theorem 3.21 are satisfied by one of them.

Another way of relaxing the stringent conditions is to look for an analogous theorem involving covers rather than partitions in some manner analogous to the way Hartmanis and Stearn develop their concept of Set Systems. Very little work has been done in these directions so far as is known to the author. A large amount of work may have gone into the calculation of Kernels of graphs composed by means other than summing: if that is so, then the paucity of the results indicate that methods for these may be difficult to come by.

In the rest of the present chapter another method for recognizing $\bigcup_i W_i$ will be discussed that has been used successfully in literature.

9. Recognizing Forcing States Through Linear Evaluation

One often tries to recognize members of $\bigcup_i W_i$ by devising languages suitable to their description. This language may be constructed by a careful evaluation of its predicates with respect to the rules of the game (as mentioned earlier, one technique for doing this with respect to the tic-tac-toe like games will be described in detail later). Another way of constructing the language might be to use predicates which have been found useful in the game (useful in some sense) and try to construct combinations of them whose denotations hopefully coincides with $\bigcup_i W_i$.

One mode of combination of predicates that has received a lot of attention in literature can be given the general name "linear combination". In its most elementary form this coincides with the mode of combination called "combination by linear threshold gates". The predicates, in these caes, denote the equivalence classes of the Kernels of functions mapping the universe of discourse into real numbers.

Let S be a set and $\varphi: S \to R$ be a function mapping S into real numbers. The Kernel of this map is the equivalence relation $E = \varphi \circ \varphi^{-1}$ defined as follows:

$$s_1 E s_2 \text{ if and only if } \varphi(s_1) = \varphi(s_2)$$

This equivalence relation partitions S into disjoint sets, yielding one set for each real number in the range of φ. Each equivalence class is the denotation of a predicate of the form $\varphi(s) = r$. Let these predicates be represented by P_r.

Let $\Phi = \{\varphi_1, \varphi_2 \ldots \varphi_n\}$ be a finite set of functions defined on S and let $\{P_{\varphi_i r} \mid \varphi_i \in \Phi, r \in \text{range of } \varphi_i\}$ be the set of predicates associated with them. Let $\omega_1, \omega_2 \ldots \omega_n$ be a set of reals. One can define a new function on s as a linear combination of the φ_i.

$$\Psi = \omega_1 \varphi_1 + \omega_2 \varphi_2 + \ldots + \omega_n \varphi_n$$

If the range of each function $\varphi_i \in \Phi$ is finite, the Kernel of Ψ defines a partition of S which is of finite index and whose equivalence classes are obtained from the equivalence classes of the Kernels of φ_i by means of set operations.

Let each of the φ_i be a characteristic function of some subset of S. Also, let us define a subset T of S as follows:

$$s \in T \text{ if } \Psi(s) \geq \theta$$

where is a specific real number. Clearly T is the union of a set of equivalence classes in the Kernel of Ψ. The characteristic function of T is often called a "linearly separable function" of the subsets defined by the φ_i and their complements.

In what follows, attention will be restricted to linear combinations of predicates in general; the discussion above is included to indicate the our understanding of linear combination of predicates extends no further than what is understood about linearly separable functions.

In what follows it will be shown how a certain function Ψ can be defined from the set of situations to reals in such a way that $\Psi(s)$ exceeds a constant value for all members of $\bigcup_i W_i$. Remarks will then be made regarding the feasibility of constructing $\Psi(s)$ as a linear combination of the other functions.

Let $\langle S,F,G,W,L\rangle$ be a board game. Define a function:

$$\Psi: S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g \rightarrow R$$

having the following property

L1) $s \in W$ and $\Psi(s') \geq \Psi(s)$ implies $s' \in W$

L2) $s \in L$ and $\Psi(s') \leq \Psi(s)$ implies $s' \in L$.

It is clear that

<u>Lemma 3.22</u> s W (s) s L (s).

Proof: Otherwise there exist real numbers y and y' such that

$$\sup_{s \in W} \varphi(s) < y < y' < \inf_{s \in L} \varphi(s).$$

Then there exist $s \in W$ and $s' \in L$ such that $y = \varphi(s)$ and $y' = \varphi(s')$. But then by L1 and L2, both s and s' are members of L and W. Since L and W are disjoint, this is impossible.

It can also be seen easily that if $S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g \neq W \cup L$, then for any situation $s \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g - W - L$,

$$\sup_{s \in W} \varphi(s) > \varphi(s) > \inf_{s \in L} \varphi(s)$$

whence in this case $\sup_{s \in W} \varphi(s) > \inf_{s \in L} \varphi(s)$. However, it is always true that for any element $s \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$, $\varphi(s) > \sup_{s \in W} \varphi(s)$ implies $s \in W$. If F and G are finite, one can extend φ into ψ, defined over some elements of $(\bigcup_{f \in F} S_f) \cup (\bigcup_{g \in G} S_g)$ as follows

$$\psi(s) = \varphi(s) \text{ if } s \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$$

$$\psi(s) = \min\{\psi(g(s)) \mid s \in S_g\} \text{ if } s \in \bigcup_{g \in G} S_g$$

$$\psi(s) = \max\{\psi(f(s)) \mid s \in S_f\} \text{ if } s \in \bigcup_{f \in F} S_f$$

In the second and third equations above, if the right hand side is not defined, then the left hand side is not defined either. Hence $\psi(s)$ may not have S as its domain. The following however, is true.

Theorem 2.23 In a progressively bounded board game where $\varphi(s') = \sup_{s \in W} \varphi(s)$ for some $s' \in W$, $s \in \bigcup_{f \in F} S_f$ and $\psi(s) \geq \sup_{s \in W} \psi(s)$ if and only if $s \in \bigcup_i W_i$.

Proof: Define a control strategy Q_F as follows:

i) If $\psi(s)$ is not defined then $Q_F(s) = f$ where f is the first element of F (in some given ordering) such that $s \in S_f$.

ii) If $\psi(s)$ is defined, then $Q_F(s)$ is the first element of F (in the given ordering) such that $\psi(Q_F(s)(s)) = \psi(s)$. By definition of ψ, such an element must exist.

Q_F is a winning strategy for all elements $s \in \bigcup_{f \in F} S_f$ such that $\psi(s) \geq \mathrm{glb}_{s \in W} \varphi(s)$. To see this, let Q_G be an arbitrary disturbance strategy. Let $\psi(s_0) \geq \mathrm{glb}_{s \in W} \varphi(s)$. Define a sequence of situations $s_0, s_1, \ldots$ as follows:

$$s_{i+1} = Q_F(s_i)(s_i) \text{ if } i \text{ is even}$$

$$s_{i+1} = Q_G(s_i)(s_i) \text{ if } i \text{ is odd.}$$

One can see immediately that for all i, $\psi(s_i) \geq \varphi(s_0)$. This is true for $s = 0$. Let it be true for $i \leq k$. If k is odd, then $s_{k+1} = g(s_k)$ for some $g \in G$. By definition of ψ, $\psi(g(s_k)) \geq \psi(s_k)$. If k is even, then by definition of Q_F, $\psi(s_{k+1}) = \psi(s_k)$.

Since the game is progressively bounded there is a last element s_m of the sequence $s_0, s_1, \ldots, s_m \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$. Also $\psi(s_m) = \varphi(s_m) \geq \psi(s_0)$ $\mathrm{glb}_{s \in W} \varphi(s)$. Hence $s_m \in W$. Since Q_G is arbitrary, Q_F is a winning strategy for s_0. Since a winning strategy exists for s_0, $s_0 \in \bigcup_i W_i$.

Conversely, let $s_0 \in W_k \subseteq \bigcup_i W_i$. If $k = 1$, then there exists an $f \in F$ such that $f(s_0) \in W$ and hence $\psi(f(s_0)) \geq \mathrm{glb}_{s \in W} \varphi(s)$. By definition of ψ, $\psi(s_0) \geq \varphi(f(s_0)) \geq \mathrm{glb}_{s \in W} \varphi(s)$. Hence if $s_0 \in W_1$, $\psi(s_0) \geq \mathrm{glb}_{s \in W} \varphi(s)$.

Let it be true that if $s \in \bigcup_{i=1}^{k} W_i$, then $\psi(s) \geq \mathrm{glb}_{s \in W} \varphi(s)$. Let $s_0 \in W_{k+1}$. Then there exists an $f \in F$ such that for all $g \in G$ such that $f(s_0) \in S_g$, $g(f(s_0)) \in \bigcup_{k=1}^{k} W_i$, $\psi(g(f(s_0))) \geq \mathrm{glb}_{s \in W} \varphi(s)$. Hence $\psi(f(s_0)) = \min\{\psi(g(f(s_0))) \mid (f(s_0) \in S_g\} \geq \mathrm{glb}_{s \in W} \varphi(s)$. But $\psi(s_0) \geq \psi(f(s_0))$ by definition. Hence $\psi(s_0) \geq \mathrm{glb}_{s \in W} \varphi(s)$.

The above theorem shows that if $\psi(s)$ could be calculated for all s for which ψ is defined, then a cautious strategy could be applied for the choice of controls. However, $\psi(s)$ cannot be calculated from definition with any practicable degree of efficiency.

In case one can easily calculate a set of functions $\varphi_1, \varphi_2 \ldots \varphi_n$ mapping S into reals such that

$$\psi = \omega_1\varphi_1 + \omega_2\varphi_2 + \ldots + \omega_n\varphi_n$$

then the predicate

$$\psi(s) \geq \theta$$

is a linear combination of the predicates corresponding to the equivalence classes of the Kernels of the functions $\{\varphi_i\}$.

Given a set of functions $\varphi_1, \varphi_2, \ldots \varphi_n$, a calculation of ψ would involve the search for a set of real numbers $\omega_1, \omega_2, \ldots \omega_n$, with the two following properties. For all $s \in S - \bigcup_{f \in F} S_f - \bigcup_{g \in G} S_g$

$\sum \omega_i \varphi_i(s) \geq \theta$ if and only if $s \in W$ and for all $s \in \bigcup_{f \in F} S_f$

$\sum \omega_i \varphi_i(s) = \max_{f \in F} \min_{g \in G} \{\sum \omega_i \varphi_i(g(f(s)) \mid s \in S_f$ and $f(s) \in S_g\}$. In the case where the φ_i are characteristic functions, methods are known for obtaining the φ_i by an adaptive procedure when they exist [25] so that they satisfy the first of the above two conditions. Some of the algorithms also indicate impossibility of fulfilling the conditions when no set $\omega_1, \omega_2 \ldots$ exists which can fulfil it. Very little theoretical study has gone into methods when no set $\omega_1, \omega_2 \ldots$ exists which can fulfil it. Very little theoretical study has gone into methods for fulfilling the second condition even when it can be fulfilled.

However, some excellent case studies have been done by Samuel [9] on the game of checkers where certain adaptive techniques have been explored for the calculation of the ω_i. The φ_i's were calculated by giving suitable mathematical interpretations to certain well-known important evaluations of checker-board positions. The ω_i's were calculated over a course of many games by adjusting them to fulfill the second condition above. The strategies resulting from the

approximate descriptions of $\bigcup_i W_i$ so obtained have yielded an extremely powerful checker-playing program. There are indications that by the use of more than one "layer" of threshold logic, a stronger program can be obtained. However, the only method available for testing these strategies seems to be operational, to wit, accumulating statistics regarding the performance of the program against strong players.

REFERENCES

1. E. A. Feigenbaum, The Simulation of Verbal Learning Behavior, Proceedings of the Western Joint Computer Conference, 19(1961), 121.

2. F. M. Tonge, Summary of a Heuristic Line Balancing Procedure, Management Science, 7(1960), 21.

3. L. R. Marino, Winning and Non-Losing Strategies in Games and Control, Report No. SRC 91-A-66-36, Case Institute of Technology, Cleveland, Ohio, 1966.

4. R. E. Bellman and S. E. Dreyfus, Applied Dynamic Programming, Princeton University Press, Princeton, N.J., 1962.

5. M. D. Mesarovic, Toward A Formal Theory of Problem Solving, Computer Augmentation of Human Reasoning (Sass and Wilkinson, Ed.), Spartan Books, Washington, D.C., 1965, 37.

6. T. G. Windeknecht, Problem Solving and Finite Automata, A Status Report on Research in Artificial Intelligence and Linguistics, Case Institute of Technology, Cleveland, Ohio, 1964.

7. G. W. Ernst and A. Newell, Some Issues of Representation in a General Problem Solver, AFIPS Conference Proceedings, 30 (1967), 583.

8. J. Von Neumann and O. Morgenstern, Theory of Games and Economic Behavior, Princeton University Press, Princeton, N.J., 1947.

9. A. L. Samuel, Some Studies in Machine Learning Using the Game of Checkers, IBM Journal of Research and Development 3(1959), 210.

10. A. Newell and H. A. Simon, The Logic Theory Machine, IRE Transactions on Information Theory IT-2(1956), 61.

11. W. W. R. Ball, Mathematical Recreations and Essays, McMillan and Co., Ltd., London, 1940, 303.

12. A. Hormann, Programs for Machine Learning, Part II, Information and Control, 7(1964), 55.

13. P. M. Cohn, Universal Algebra, Harper & Row, New York, 1965.

14. A. Newell, J. C. Shaw and H. Simon, Report on a General Problem Solving Program, Proceedings of the International Conference on Information Processing, UNESCO, Paris, 1959, 256.

15. S. Amarel, On Machine Representation of Problems of Reasoning About Actions - The Missionaries & Cannibal Problem, RCA Laboratories, Princeton, N.J., 1966.

16. A. Church, Introduction to Mathematical Logic, Princeton University Press, Princeton, N.J., 1956.

17. S. T. Hu, Threshold Logic, University of California Press, Berkeley, 1965.

18. R. L. Ashenhurst, The Decomposition of Switching Functions, Proceedings of the International Symposium on the Theory of Switching, Harvard University Press, Cambridge, Massachusetts, 1959.

19. J. S. Bruner, J. J. Goodnow and G. A. Austin, A Study of Thinking, John Wiley & Sons, New York, 1956.

20. I. H. Sublette, Recognition of Class Membership by Means of Weak, Statistically Dependent Features, Radio Corporation of America, Princeton, N.J., 1966.

21. A. Lehman, A Solution for the Shannon Switching Game, U.S. Army Mathematical Research Center Tech. Summary Report 308; July 1962.

22. R. G. Busacker and T. L. [illegible]asy, Finite Graphs and Networks, McGraw Hill Book C[illegible], [illegible]ew York, 1965.

23. C. Berge, The Theory of Graphs and its Applications, John Wiley and Sons, New York, 1962.

24. J. Hartmanis and R. E. Stearns, Algebraic Structure Theory of Sequential Machines, Prentice Hall, Englewood Cliffs, N.J., 1966.

25. F. Rosenblatt, Perceptron Experiments, Proceedings of the IRE, 48,(1960), 301.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Systems Research Center Case Western Reserve University Cleveland, Ohio 44106	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

SOME RESULTS IN A THEORY OF PROBLEM SOLVING

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

SRC report

5. AUTHOR(S) (First name, middle initial, last name)

Banerji, Ranan B.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
November 1967	151	25

8a. CONTRACT OR GRANT NO. AF-OSR-125-67, and AF-OSR-125-65 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) SRC 126-A-67-59 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) Also to be published jointly by RCA Research Lab

10. DISTRIBUTION STATEMENT

~~Available upon request through:~~ System[redacted]
[redacted] Reserve University
Cleve[redacted]

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

A general model of a control problem has been specialized to yield models of one-person games (problems) and two-person extended form games. It has been indicated that the practical application any solution strategy needs succinct descriptions of certain subsets of the state space of the control problem. The idea of description has been formalized and the importance of description languages discussed.

Certain techniques for constructing solution strategies have been discussed and the sets associated with their application precisely identified.

DD FORM 1 NOV 65 1473

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
SOME RESULTS IN A THEORY OF PROBLEM SOLVING						

Security Classification